Modicon Quantum

Hot Standby System
User Manual

07/2011

35010533.08

www.schneider-electric.com

Schneider Electric

The information provided in this documentation contains general descriptions and/or technical characteristics of the performance of the products contained herein. This documentation is not intended as a substitute for and is not to be used for determining suitability or reliability of these products for specific user applications. It is the duty of any such user or integrator to perform the appropriate and complete risk analysis, evaluation and testing of the products with respect to the relevant specific application or use thereof. Neither Schneider Electric nor any of its affiliates or subsidiaries shall be responsible or liable for misuse of the information contained herein. If you have any suggestions for improvements or amendments or have found errors in this publication, please notify us.

All pertinent state, regional, and local safety regulations must be observed when installing and using this product. For reasons of safety and to help ensure compliance with documented system data, only the manufacturer should perform repairs to components.

When devices are used for applications with technical safety requirements, the relevant instructions must be followed.

Failure to use Schneider Electric software or approved software with our hardware products may result in injury, harm, or improper operating results.

Failure to observe this information can result in injury or equipment damage.

Table of Contents

Safety Information

Important Information

NOTICE

Read these instructions carefully, and look at the equipment to become familiar with the device before trying to install, operate, or maintain it. The following special messages may appear throughout this documentation or on the equipment to warn of potential hazards or to call attention to information that clarifies or simplifies a procedure.

The addition of this symbol to a Danger or Warning safety label indicates that an electrical hazard exists, which will result in personal injury if the instructions are not followed.

This is the safety alert symbol. It is used to alert you to potential personal injury hazards. Obey all safety messages that follow this symbol to avoid possible injury or death.

DANGER

DANGER indicates an imminently hazardous situation which, if not avoided, **will result in** death or serious injury.

WARNING

WARNING indicates a potentially hazardous situation which, if not avoided, **can result in** death or serious injury.

⚠ CAUTION

CAUTION indicates a potentially hazardous situation which, if not avoided, **can result in** minor or moderate injury.

CAUTION

CAUTION, used without the safety alert symbol, indicates a potentially hazardous situation which, if not avoided, **can result in** equipment damage.

PLEASE NOTE

Electrical equipment should be installed, operated, serviced, and maintained only by qualified personnel. No responsibility is assumed by Schneider Electric for any consequences arising out of the use of this material.

A qualified person is one who has skills and knowledge related to the construction and operation of electrical equipment and its installation, and has received safety training to recognize and avoid the hazards involved.

About the Book

At a Glance

Document Scope

This guide describes the Modicon Quantum Hot Standby S908 System and Quantum Ethernet I/O Ethernet system consisting of:

- Unity Pro software
- Modicon Quantum Hot Standby CPUs:
  - 140 CPU 671 60
  - 140 CPU 671 60S
  - 140 CPU 672 61
- power supplies
- remote I/O (RIO) network
- distributed I/O (DIO)
- RIO Head modules

This guide describes how to build a Quantum Hot Standby system. Users of Concept/ProWORX Quantum Hot Standby systems please note that there are significant differences that exist between Unity Pro and legacy systems, and where important, this guide identifies those differences.

Who should use this document?

Anyone who uses a Hot Standby system or needs fault-tolerant availability through redundancy in an automation system.

You should have knowledge of programmable logic controllers (PLCs). Familiarity with automation controls is expected.

You should possess a working knowledge of the Unity Pro software. It is helpful if you are familiar with Ethernet networks.

Validity Note

This document is valid from Unity Pro 6.0.

Related Documents

Title of Documentation	Reference Number
Quantum Ethernet I/O Ethernet Remote I/O Modules Installation and Configuration Guide	S1A48978 (English), S1A48981 (French), S1A48982 (German), S1A48983 (Italian), S1A48984 (Spanish), S1A48985 (Chinese)
Quantum Ethernet I/O Global System Planning Guide	S1A48959 (English), S1A4896 (French), S1A48962 (German), S1A48964 (Italian), S1A48965 (Spanish), S1A48966 (Chinese)
Modicon Quantum Change Configuration on the Fly User Guide	S1A48967 (English), S1A48968 (French), S1A48969 (German), S1A48970 (Italian), S1A48972 (Spanish), S1A48976 (Chinese)
Unity Pro Program Languages and Structure Reference Manual	35006144 (English), 35006145 (French), 35006146 (German), 35006147 (Spanish), 35013361 (Italian), 35013362 (Chinese)
Unity Pro Operating Modes	33003101 (English), 33003102 (French), 33003103 (German), 33003104 (Spanish), 33003696 (Italian), 33003697 (Chinese)

Quantum with Unity Pro Hardware Reference Manual	35010529 (English), 35010530 (French), 35010531 (German), 35010532 (Spanish), 35013975 (Italian), 35012184 (Chinese)
Unity Pro Installation Manual	35014792 (French), 35014793 (English), 35014794 (German), 35014795 (Spanish), 35014796 (Italian), 35012191 (Chinese)

You can download these technical publications and other technical information from our website at www.schneider-electric.com.

Product Related Information

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

The application of this product requires expertise in the design and programming of control systems. Only persons with such expertise should be allowed to program, install, alter, and apply this product.

Follow all local and national safety codes and standards.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

User Comments

We welcome your comments about this document. You can reach us by e-mail at techcomm@schneider-electric.com.

Introducing the Modicon Quantum Hot Standby System

I

Modicon Quantum Hot Standby System

1

Overview

This chapter briefly describes the Modicon Quantum Hot Standby system and some of the concepts needed to understand the system.

Also, included is information about the Hot Standby Safety system (only available with S908 RIO) and compatible equipment.

What's in this Chapter?

This chapter contains the following sections:

1.1 Quantum Hot Standby Introduction

Overview

This section describes information you need to know before starting to configure and operate a Quantum Hot Standby system.

What's in this Section?

This section contains the following topics:

Terminology

Hot Standby Terms

This manual uses many technical terms and acronyms. Some of the most commonly used are:

- *Application Program*:
  This is the software program (user logic) you write to provide monitoring and control for your application.
- *Copro*:
  This term is short for Coprocessor. This manual uses the term Copro specifically to refer to the coprocessor that governs the exchange of data between the Hot Standby PLCs through the Sync-link between the CPUs.
- *Extended rack*:
  This rack contains I/O modules and is connected to the main rack through Rack Expender modules and their cable. Its modules are considered to be in the same backplane as the main rack.
- *In-rack I/O*:
  This term refers to any I/O that is directly connected to the Quantum rack's backplane (in the main rack or an extended rack), without any intervening fieldbus connections and, therefore, includes all analog and discrete I/O modules. In-rack I/O is locally and non-redundantly managed ("Local In-rack I/O" or just "Local I/O") in the first section of the MASK task of the Hot Standby application program.
- *Local PLC/Peer PLC:*
  The Local PLC is the PLC in the Hot Standby system that you are working on. The other PLC is the Peer PLC. The local PLC can be the Primary or Standby and the Peer PLC can be the Standby or Primary.
- *Main rack*:
  This is the rack that supports the processors (CPU module and RIO Drop adapter modules (140_CRA_93•_00 or 140 CRA 312 00) and I/O modules.
- *Program cycle*:
  In a Hot Standby system, the program cycle can only use the MAST task (see Exclusive Use of MAST Task *(see page 49)*. The MAST task cycle is classically divided in the four main parts:
  - input drivers: all input modules associated with the MAST Task are scanned
  - Hot Standby system functions: data exchange between CPU and Copro and system checks
  - user logic execution: the system executes the application program relative to the MAST Task
  - output drivers: the system applies the outputs evaluated during the application program execution to all the output modules associated with the MAST Task

- *Switchover*:
  This refers to the moment when application control transfers from the Primary controller to the Standby controller. The Switchover event has a finite duration. It can be initiated:
  - manually
  - by the application program
  - automatically by system conditions

Purpose and Features

Purpose

The Quantum Hot Standby is an industrial control platform intended to provide automatic redundancy for a wide range of conditions. The main components of the system are two PLCs called the main or "Primary" PLC and the secondary or “Standby” PLC, which has an identical configuration as the Primary controller.

By being programed to detect and respond to defined system conditions, the Quantum Hot Standby system can automatically transition from the Primary controller (and its associated modules) to the Standby controller (and its identical modules). This transition, called the "Switchover", takes place in a short time (the length of the watchdog plus one program cycle).

Because the Quantum Hot Standby detects and responds automatically to a wide range of detected error conditions, you do not need to manage these detected errors in your application programs.

For a more complete redundancy to increase the availability of the system, dual drops, dual cabling, dual sensors and actuators can be used with a Quantum Hot Standby configuration.

Features

The Quantum Hot Standby system:
- increases the system availability of your treatment plants and remote stations, allowing you to conduct many maintenance operations while the system is operational
- is a single-detected-fault-tolerant system, that is, the system can continue operating even though one component of the system is inoperative
- provides control redundancy for (Quantum Ethernet I/O Ethernet or S908 Remote I/O systems)
- requires no specialized modules or equipment other than the Hot Standby PLCs and Ethernet modules. You can use standard Quantum racks, power supplies, and I/O modules (analog and discrete).
- offers a user-friendly development environment compatible with IEC 6113 - 3
- allows creation of a redundant-ready application program almost as easily as for a standalone PLC and requires few changes from your normal programming methods

NOTE: If not mentioned in the document, all features of Standalone High End Quantum PLCs are available in Quantum Hot Standby PLCs.

Overview

Quantum Hot Standby

The Quantum Hot Standby controller implements system redundancy using redundant hardware and by automatically switching over to the Standby (backup) hardware when certain defined system events are detected. While your prior PLC experience is very important to the proper use of this system, you need to become familiar with new concepts, practices, and restrictions to properly implement and manage the Quantum Hot Standby's redundancy.

NOTE: Users of Premium Hot Standby, Quantum legacy or other redundant systems should be aware that differences exist between the redundancy provided by these systems and that provided by the Quantum Hot Standby system. The differences include terminology, the conditions for switching to the standby system, system requirements and restrictions, etc.

Redundant Hardware

Two Controllers: Primary and Standby

The basic requirement for a Quantum Hot Standby system is to use two completely identical Hot Standby PLCs of one of the following types:
- 140 CPU 671 60
- 140 CPU 672 61
- 140 CPU 671 60S (only available for a S908 RIO)

These controllers must have the same firmware versions and be positioned in the same slots on their respective Quantum racks. They must also run the same application program.

In a system that is operating nominally, with both controllers fully functional, the two identical controllers assume one of two operating modes:
- One controller acts as the Primary PLC, operating in the “Run Primary” mode.
- The other controller acts as the Standby PLC, operating in the “Run Standby” mode.

The role of the Primary PLC is almost identical to that of a standalone PLC in a non-redundant system. It runs your entire application program and, thereby, provides the normal control functions you would expect from a standalone PLC.

The Primary CPU controller:
- executes the whole application program (first section of the MAST task included)
- controls the Remote I/O
- updates the Standby CPU controller every scan (program cycle)

The major differences of the Primary CPU from a standalone PLC are:
- The Primary Hot Standby controller communicates regularly with its Standby PLC so that the Standby remains ready to assume the Primary role if required.
- The Primary PLC monitors itself and certain associated equipment for specific conditions that dictate a Switchover to the Standby controller.

The role of the Standby PLC is different from a standalone PLC. Its role is to remain ready to assume control of the system at a moment’s notice and yet not interfere with the control asserted by the Primary controller. To do so, it must regularly receive application data and I/O states computed by the Primary controller.

The Standby CPU:
- executes only the first section of the application program MAST task
- verifies the availability of the Primary CPU and CRP modules
- can update the Primary CPU about the status of its CPU, CRP modules and Drop connections
- does not control the Remote I/O

The Standby PLC also regularly communicates information back to the Primary PLC using a group of System Words: the Reverse Transfer Registers. The content of these System Words is configurable. The most common use is to provide the Primary PLC application program information about the health of the Standby controller and its associated modules.

Distinguishing Between Controllers

The two physical controllers are assigned as either PLC A or PLC B. This assignment is used to configure the IP address of the CRP RIO Head modules.

Distinguishing between the A and B Hot Standby CPUs allows:
- assigning a physical location to each CPU
- defining which CPU is the Primary at system start-up

⚠ WARNING
UNINTENDED EQUIPMENT OPERATION
Confirm the A/B assignment of a PLC before taking any action on it.
Failure to follow these instructions can result in death, serious injury, or equipment damage.

Never assume that a PLC is in a certain operating mode before installing, operating, modifying, or servicing it. The operating modes of both Hot Standby PLCs can be determined by viewing their LCD keypads, LEDs and System Status Words.

NOTE: In a Hot Standby system, the CRP IP addresses are not swapped during a Switchover.

At the first startup of a CPU without the A/B assignment the `Hot Standby` menu is displayed on the keypad LCD allowing the user to assign A or B to the CPU.

The user can also assign/change the A/B assignment of the Hot Standby CPUs using the keypad LCD. After modification the CPUs reset its CRP RIO Head modules.

NOTE: When a CPU is in the RUN mode its A/B assignment cannot be changed. It must be in the STOP mode to change its assignment.

The two CPU cannot have the same A or B assignment:
- If a CPU starts with the same assignment as the other CPU, this CPU goes to the STOP mode, displays the `Hot Standby` menu and waits for an assignment from the keypad.
- If you replace one of the PLCs, the identification of PLC A and PLC B may no longer align with the Primary and Standby operating modes.
  The same thing is true for any physical labels you might apply to your PLCs to distinguish them in your system.

CRP Head modules IP addresses are based on the user configured IP address in Unity Pro and the A/B assignment.

Establishing the Primary and Standby Controllers

If the system is properly configured, the first Hot Standby PLC to which power is applied assumes the role of the Primary controller. Therefore, you can determine controller roles by delaying the application of power to one PLC using a time-lag relay or some related means.

When you apply power simultaneously to two properly configured Hot Standby PLCs, the firmware automatically assigns the role of the Primary controller based on the A/B assignment. The PLC that is A becomes the Primary controller.

Identical RIO Head Modules Required

In addition to requiring two identical controllers, a Quantum Hot Standby system requires a minimum of two identical Quantum RIO “Head” modules, one on each rack.

These two modules can be:
- 140 CRP 931 00 (for S908 I/O Drops)
- 140 CRP 932 00 (for S908 I/O Drops)
- 140 CRP 312 00 (for Ethernet I/O Drops)

Like the controllers, the rack positions and firmware versions of the CRP modules must be identical.

CPU-Sync Link

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

- Make an uninterrupted point-to-point connection between the Hot Standby CPU-sync link ports.
- Do not connect any other Ethernet devices on the same network cabling as the CPU-sync link.
- Do not exceed maximum Ethernet cable lengths for the type of cable selected.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

The CPU-sync link is the main communications channel for providing Quantum Hot Standby redundancy. It is located between the Hot Standby (labeled “HSBY Link”) ports on the face of each controller. Do not include switches and hubs on this link. Refer to Hot Standby Sync-Link *(see page 27)* for details.

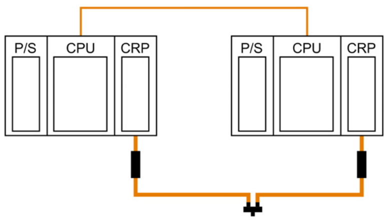
RIO Redundant Link

A RIO network (S908 or Ethernet) is used as a redundant link for the Hot Standby system. This redundant link is mandatory for some operating modes and error detection.

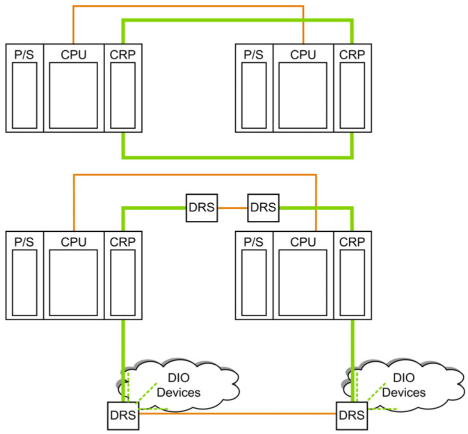
Either an S908 or Ethernet RIO CRP Head module can be used for the RIO redundant link.

If S908 or Ethernet RIO Drop are not needed in a Hot Standby system (for a DIO only system), you must still install S908 or Ethernet CRP Head modules and their connecting network (just as you would for a RIO system).

S908 system without any CRA Drops:



Ethernet systems without CRA Drops:



In addition to the Hot Standby Sync-Link *(see page 27)*, an Ethernet system has two types of connections between the CRPs that can use ConneXium extended managed switches (called dual ring switches DRSs in this architecture) for:

- one side of the ring, a connection with a maximum of two DRSs for a long distance connection *(see page 28)* (no Remote Drops or Distributed I/O devices are allowed), which can be connected between the CRPs over long distances with fibre optic cable
- for the other side of the ring, Remote Drops or DIO devices (“DIO Clouds”) are allowed

For more information, refer to Dual Ring Switches *(see page 37)*.

Core Hot Standby Hardware

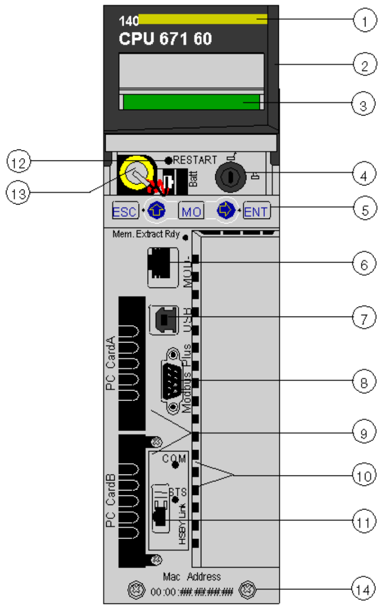
The basic requirements for Quantum Hot Standby system is two identical:

- Hot Standby CPUs
- sync link between the CPUs (refer to Identical RIO Head Modules *(see page 23)*
- CRP modules and their connecting network
- racks
- power supplies

Quantum Hot StandBy CPU Front Panel

Front Panel

The figure shows an Hot StandBy CPU module front panel:



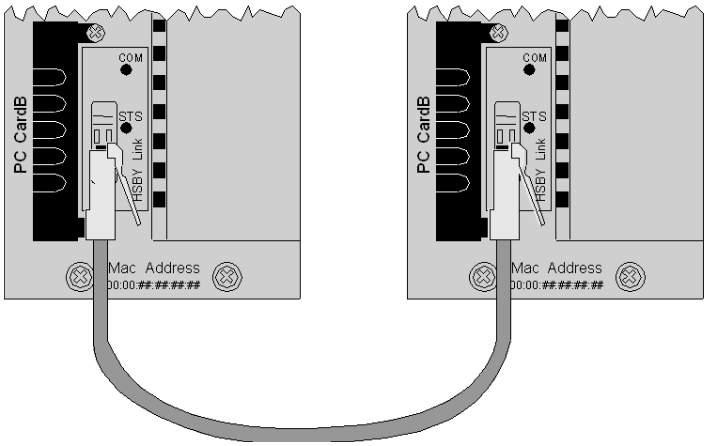
1 Model number, module description, color code
2 Lens cover (open)
3 LCD Display (here covered by lens cover)
4 Key switch
5 Keypad (with 2 red LED indicators)
6 Modbus port (RS-232) (RS-485)
7 USB port
8 Modbus Plus port
9 PCMCIA slots A and B
10 LED indicators (yellow) for Ethernet communication
11 HSBY Link fiber optic communication port
12 Reset button
13 Battery (user installed)
14 2 screws

NOTE: Quantum CPU are equipped with two receptacles for Schneider PCMCIA cards (other cards are not accepted).

Hot Standby Sync-Link

Cable Connections

The Copros in the Quantum Hot Standby CPUs must be connected by a crossed fiber optic cable plugged into the **HSBY Link** socket:



If the cable is not connected properly, the Quantum Hot Standby processors cannot communicate, and the Hot Standby system cannot function.

The fiber optic cables are sold separately:

Multi mode Models for 140 CPU 671 60	**Description**
490NOR00003	3 m MTRJ/MTRJ
490NOR00005	5 m MTRJ/MTRJ
VDIF0646463505	15 m MTRJ/MTRJ
Single mode Models for 140 CPU 672 61	
490NOL10005	5 m LC/LC

The fiber connection between Primary CPU and Standby CPU must be a direct cable connection, which reduces the components that could become inoperative in the redundant system.

NOTE: Refer to the recommendations *(see page 206)* for fiber optic cable use.

⚠ WARNING

UNEXPECTED EQUIPMENT OPERATION

Do not use hubs and switches as part of the fiber optic link.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Interrupted Sync-Link

If there is break in the commutation over the Sync-link cable:
- The Primary CPU detects the link failure and remains the Primary CPU.
- The Standby CPU requests the Standby CRP if the Primary CPU exists.
- The Standby CRP acknowledges that the Primary CPU does exist.
- The Standby CPU goes Offline.

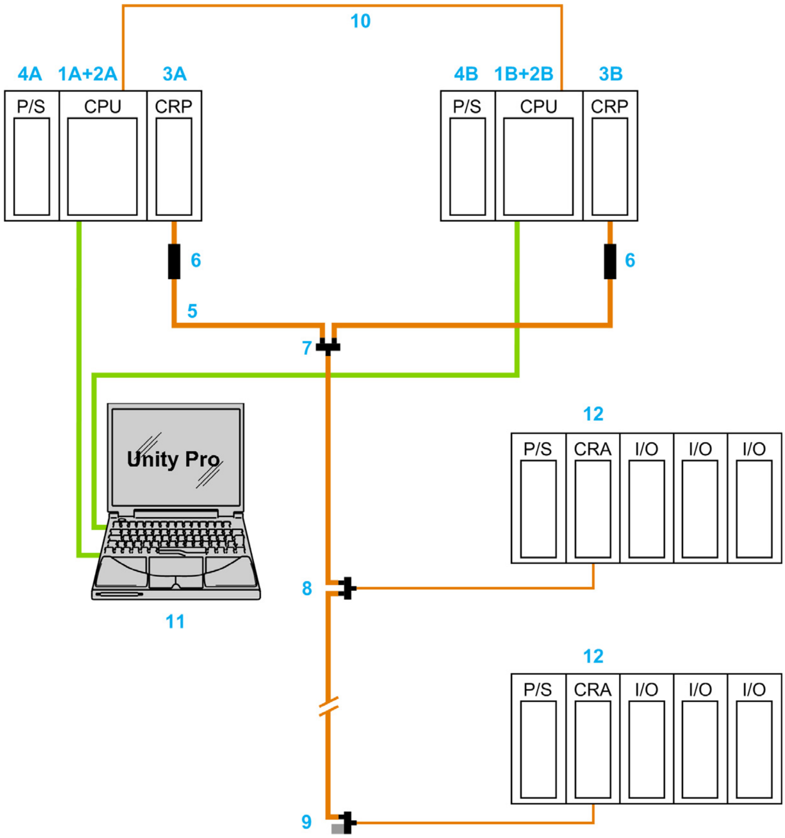
Connecting Two Backplanes

The Primary CPU and Standby CPU backplanes may be placed as much as:
- 4 km (2.5 miles) for the 140 CPU 671 60 and 140 CPU 671 60S
- 16 km (10 miles) apart for the 140 CPU 672 61

S908 Hot Standby Hardware and Topology

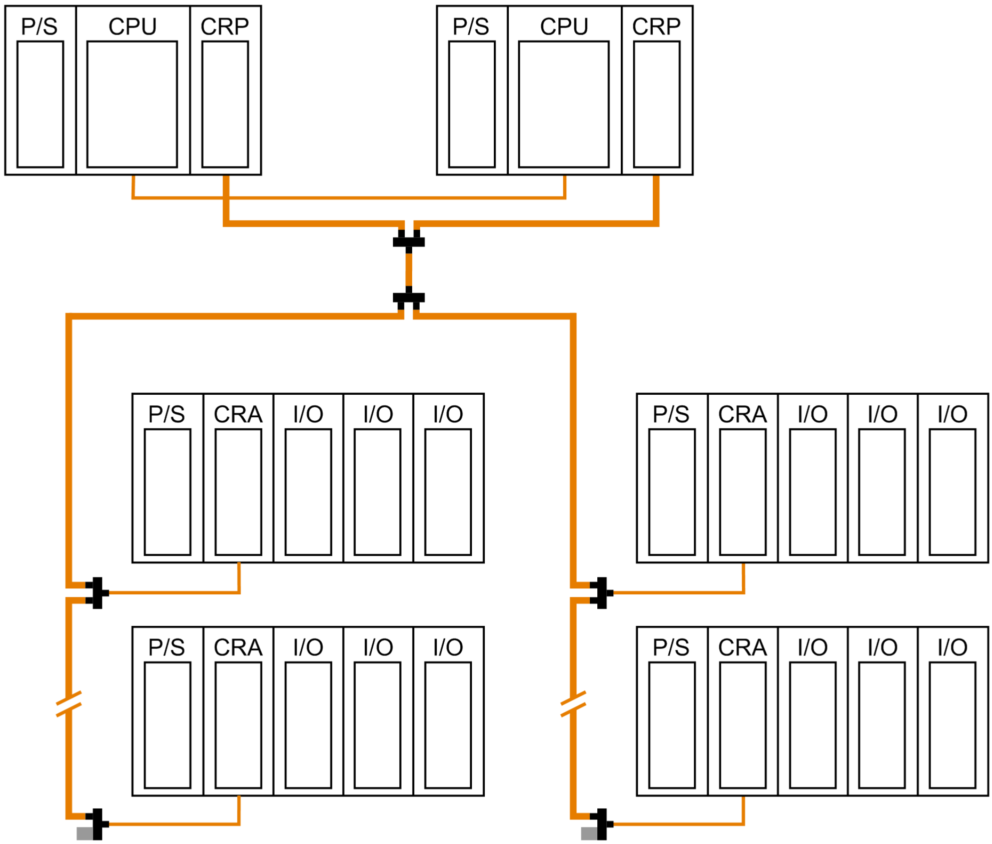
S908 System Components

The following figure shows the basic, one-bus architecture of an S908 Quantum Hot Standby system:



1A+2A CPU and Copro of Primary controller
1B+2B CPU and Copro of Standby controller
3A+3B Quantum RIO Head modules
4 Primary and Standby power supplies
5 Coaxial cable
6 Self terminating F adapter
7 Splitter
8 Tap
9 Tap withTrunk terminator
10 CPU-Sync firer optic link
11 Unity Pro workstation
12 S908 RIO Drops

The following is a two-bus S908 system:



S908 RIO Network

The 140 CRP 93* 00 RIO Head module is connected to the 140 CRA 93* 00 RIO Drop modules through cables, self-terminating F adapters, signal splitters and taps.

Dual cabling offers even more Hot Standby redundancy.

There may be up to 31 RIO drops connected to the two RIO Head modules.

The minimum Quantum Hot Standby does not require any RIO drops, but it must include at least one pair of connected RIO Head modules.

Parts List

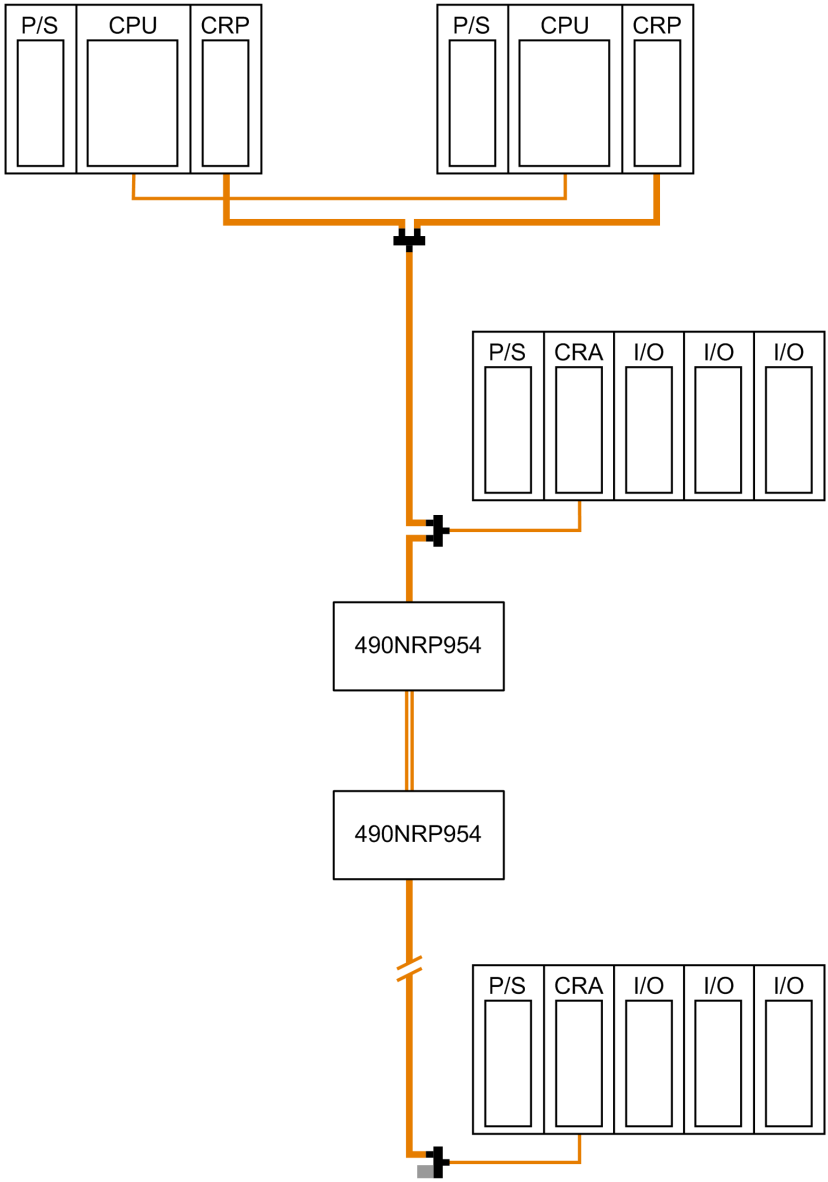
The parts list for a S908 single-cabled Hot Standby system is as follows:

Name	Reference	Minimum Firmware Version	Number of Units
Quantum Standard Racks	140 XBP 0•• 00	—	2
Quantum Power Supplies	140 CPS ••• •0	—	2
Quantum Hot Standby Controller	140 CPU 672 61 140 CPU 671 60 140 CPU 671 61 140 CPU 671 61S	— — — —	2 2 2 2
Quantum Hot Standby RIO Head Modules	140 CRP 931 00 140 CRP 932 00	2.0 2.0	2 2
Quantum Hot Standby RIO Drop Modules	140 CRA 931 00 140 CRA 932 00	See Software Requirements *(see page 40)*	As needed As needed
Self Terminating F Adaptor	52 0411 000	—	2
Splitter	MA 0186 100	—	1
Tap	MA 0185 100	—	As needed
Trunk Terminator	52 0422 000	—	As needed

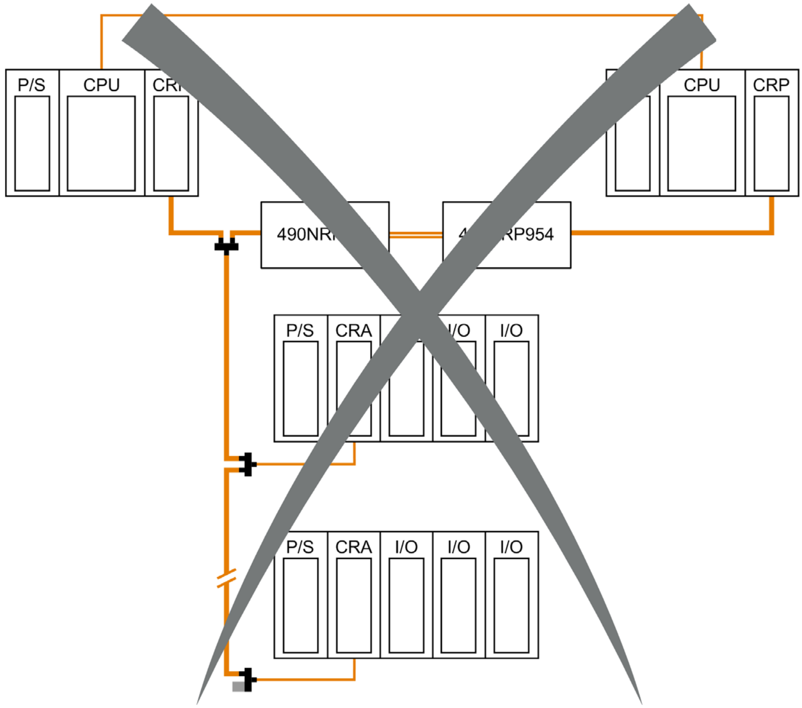
NOTE: The above hardware is always required in Hot Standby systems, but it does not provide a useful redundant system because no redundantly managed I/O modules are included.

490 NRP 954 00 Fiber Optic Repeaters

NRP repeaters are only used between S908 CRP RIO Drop modules:



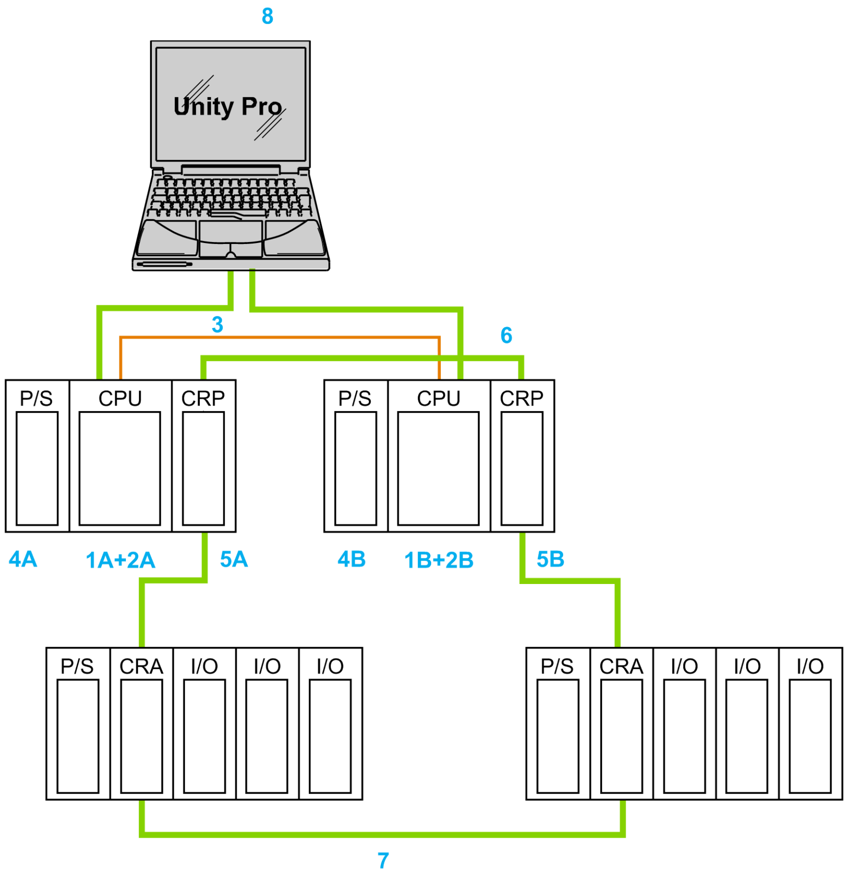
NRP repeaters cannot be used between the two Hot Standby CRP RIO Drop modules:



Quantum Ethernet I/O Hot Standby Hardware and Topology

Quantum Ethernet I/O System Components

The following figure shows the single, daisy chain ring architecture of a Quantum Ethernet I/O Hot Standby system:



1A+2A CPU and Copro of Primary controller
1B+2B CPU and Copro Standby controller
3 CPU-Sync fiber optic link
4A+4B Primary and Standby power supplies
5A+5B Primary and Standby RIO Head modules
6 Ethernet connection between RIO Head modules
7 Ethernet RIO Drops in a daisy chain configuration
8 Unity Pro workstation

Quantum Ethernet I/O RIO Network

The 140 CRP 312 00 RIO Head module is connected to the 140 CRA 312 00 RIO Drop modules through Ethernet cables and, if necessary, dual ring switches (DRSs). This network must use a daisy chain ring topology and the RSTP 2004 protocol.

There may be up to 30 RIO Drops connected to the 2 CRP modules on one side of the main ring. If more than 30 CRA Drops are needed, there can be additional Drops connected in one or more sub-rings using dual ring switches (DRSs).

The other side of the main ring must have the 2 CRP modules directly connected without RIO Drops between them. There can be a maximum of two DRSs on this side of the ring.

The minimum Quantum Hot Standby does not require any RIO drops, but it must include at least one pair of 140 CRP 312 00 Head modules.

Parts List

The parts list for a Quantum Ethernet I/O Hot Standby system is as follows:

Name	Reference	Minimum Firmware Version	Number of Units
Quantum Standard Racks	140 XBP 0•• 00	—	2
Quantum Power Supplies	140 CPS ••• •0	—	2
Quantum Hot Standby Controller	140 CPU 671 60 140 CPU 672 61	3.0 3.0	2
Quantum Hot Standby RIO Head Modules	140 CRP 312 00	1.0	2
Quantum Hot Standby RIO Drop Modules	140 CRA 312 00	1.0	As needed
Dual Ring Switch (DRS)	TCSESM083F23F1 TCSESM063F2CU1 TCSESM063F2CS1	—	As needed

NOTE: The above hardware is always required in Hot Standby systems, but it does not provide a useful redundant system because no redundantly managed I/O modules are included.

Additional Quantum Ethernet I/O Redundancy

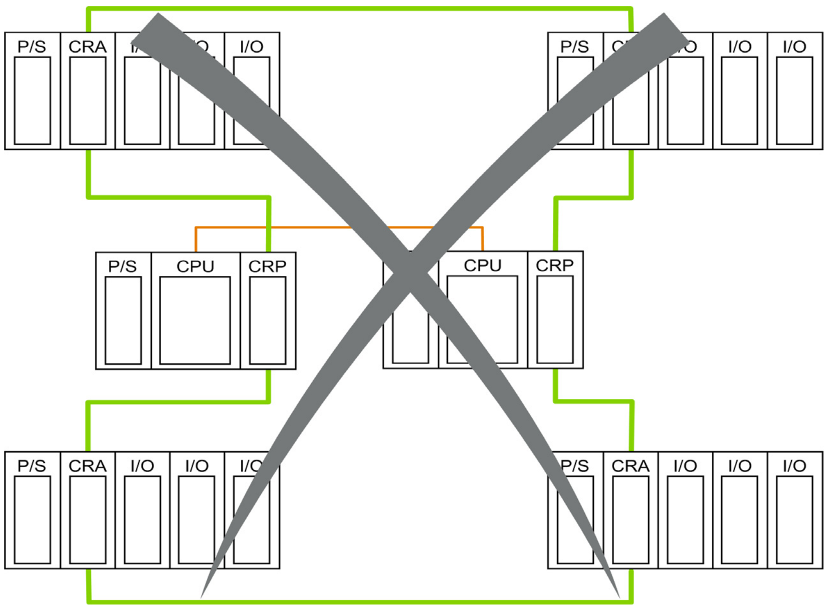
The S908 and Quantum Ethernet I/O Hot Standby systems offer redundancy by having 2 identical PLCs (Primary and Standby) to control the system I/O.

A Quantum Ethernet I/O Hot Standby system offers increased redundancy when its Remote I/O is configured in a daisy chain ring.

In this configuration there are three links *(see page 34)* between the Primary and Standby PLCs:

1. directly between the two Copros over the Sync-link *(see page 27)*
2. over the daisy chained Ethernet Remote I/O Drops between the 2 CRPs
3. over the Ethernet direct link between the 2 CRPs

This direct link cannot have Remote or Distributed I/O Drops, for example, the following is not allowed:



Dual Ring Switch

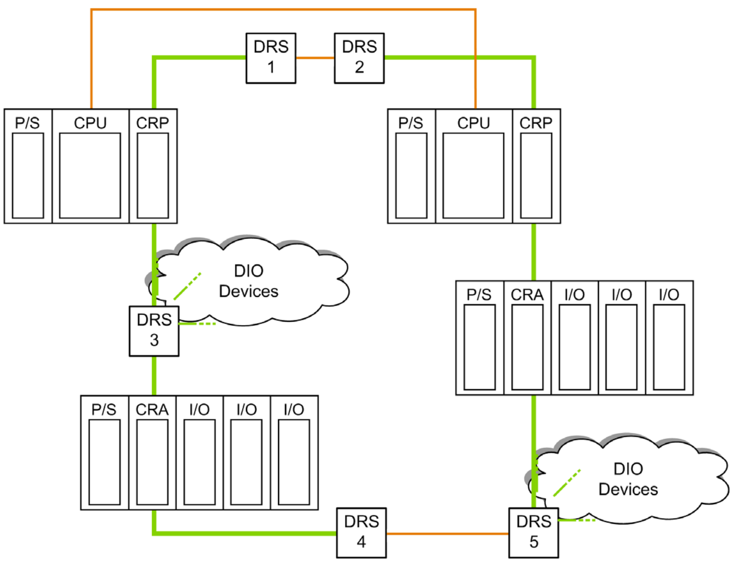
A dual ring switch (DRS) can be used to:
- insert a sub-ring into the main daisy chain ring
- connect distributed I/O devices to the system
- connect Ethernet CPR Head modules and CPA Drop modules.
  For connections:
  - < 100 m use copper cable
  - > 100 m fibre optic cable must be used

Each DRS counts as 2 devices on a ring.

For more information about the available DRSs, refer to the ConneXium Dual Ring Switch.

Dual Ring Switch Topology Examples

The following example shows 2 possible uses of DRSs:



Actions of the DRSs:

1 & 2 These DRSs connect the 2 Ethernet CRP modules in the Hot Standby PLCs using fibre optic cable because the distance is > 100 m.

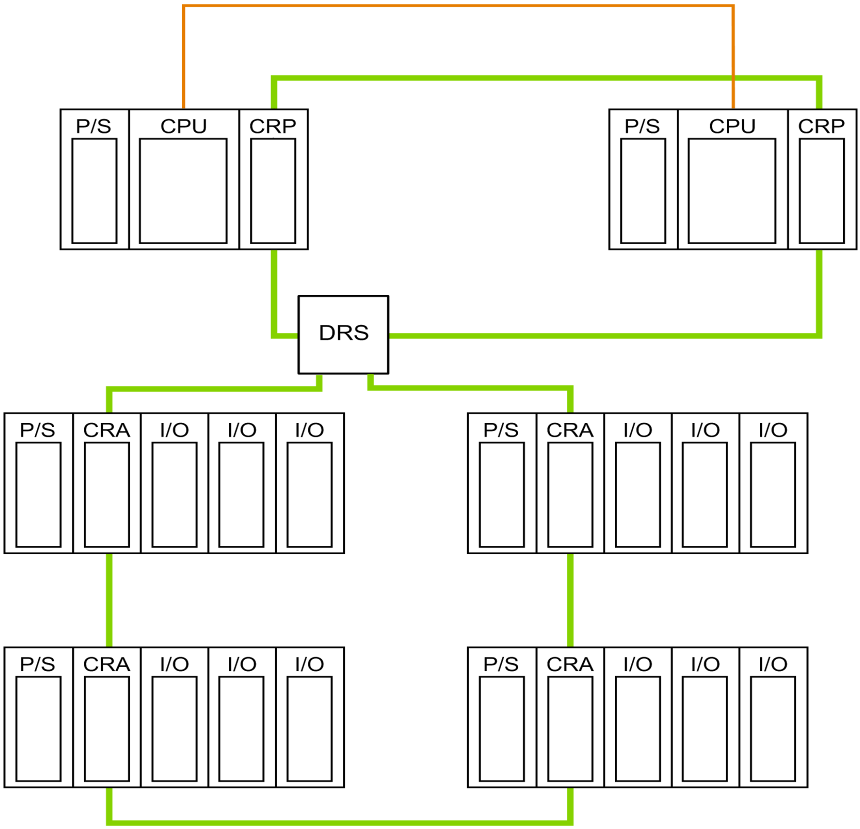
3 This DRS attaches Distributed I/O devices to the main daisy chain ring.

4 & 5 These DRSs connect the CRA modules of 2 Remote I/O Drops using fibre optic cable because the distance is > 100 m.

5 This DRS also attaches Distributed I/O devices to the main daisy chain ring.

On this ring there are 2 CRP devices, 2 CRA devices and 5 DRSs that count as 10 devices for a total 14 devices. 32 - 14 = 18 additional devices can be added to this ring.

A DRS can be used to connect a sub-ring to the main ring:



On the main ring there are 2 CRP devices, 4 CRA devices and 1 DRS that count as 2 devices for a total 8 devices. 32 - 8 = 24 additional devices can be added to the main ring.

Configuration Requirements

Identical Hardware and Software

In previous sections, we stated the requirement for identical controllers and CRP Head modules. In fact, the requirement for identical configurations extends to all equipment on both the Primary and Standby racks, and even to the application programs. To create a working Hot Standby system, you must meet all of the following hardware/firmware requirements or your system cannot come Online.

Identical Hardware

There must be identical hardware in both controllers:

- Identical Quantum Hot Standby controllers with identical CPU and Copro firmware, identical memory cards and accessories, occupying the same rack positions. You can permit differing firmware versions on a temporary basis so that operational firmware upgrades *(see page 195)* are possible.
- Identical In-rack I/O. Any In-rack I/O must be identical, including having identical firmware versions and hardware revisions (if applicable), and must occupy the same rack positions.

NOTE: Since the applications are identical in both controllers, the In-rack I/O has to be identical in both PLCs so that the Standby application can handle this I/O if it becomes the Primary controller.

- Identical module cartridges and accessories. For In-rack communication and I/O modules that accept such accessories, any cartridges used must be identical, and identically positioned and configured.
- Identical extended Quantum racks (140 XBP 004 00 through 140 XBP 016 00 backplanes). Each PLC must contain the same number of racks. The rack IDs used must be the same on each PLC.
- Identical Quantum 140 CPS ••• •• power supplies, occupying the same rack positions, and, ideally, supplied by different circuits.
- Identical cabling and cabling systems, fully shielded, and compliant with the length requirements for the type of fieldbus you employ.

Identical Software

Identical application programs and configurations must be loaded on both Quantum Hot Standby controllers in both Primary and Secondary controllers.

NOTE: You can temporarily permit different software on the two controllers so that operational software modification can be made while running, refer to the CCOTF User Manual.

Software and Firmware Requirements

For a Ouantum Hot Standby system the following software and firmware is required:

- 140 CPU 671 60 and 140 CPU 672 61firmware: 3.0
- S908 140 CRP 93• 00 firmware: 2.0
- S908 140 CRA 93• 00 firmware: 2.0
- Unity Pro 6.0 XL and XLS
- 140 CRP 312 00 firmware: 1.0
- 140 CRA 312 00 firmware: 1.0

Establishing Redundancy

Database Exchange

The Ouantum Hot Standby provides redundancy by maintaining its Standby PLC and associated modules in a state where they can assume the Run Primary operating mode quickly. This means that the Standby PLC must have all of the information necessary to mirror the I/O and data states present on the Primary PLC and that this information must be regularly updated. For the Ouantum Hot Standby, the collected information is called the "database" and the regular exchange of this database is referred to as the "database transfer".

Just after the Primary PLC finishes reading the input values it transfers the database to its Copro, which in turn transmits it over the CPU-sync link to the Standby PLC Copro. The Standby PLC then applies the information in the database as required.

The database that is cyclically transferred from the Primary PLC to the Standby PLC (via the Copros and the CPU-sync link) includes both system data and user application data and I/O. In both cases, some of this data is located (addressable) and some is unlocated. The data exchanged during every MAST task are listed below.

System Information

- LOCATED:
  - System Bits:
    %S30: activation of MAST task
    %S38: enabling/inhibition of events tasks
    %S50: clock write
    %S59: clock increase
    %S94: replace current value
    %S117: RIO Error on Ethernet I/O Network
    %S118: RIO Error on S908 I/O Network
  - System Words:
    %SW0: Set scanning period for MAST task
    %SW8 - 9: task Input/Output inhibition
    %SW49 - 53: date and time information
    %SW59: updates date and time values
    %SW60: Hot Standby Command register, refer to Hot Standby Command Register *(see page 86)*
    %SW61: Hot Standby Status register, refer to Hot Standby Status Register *(see page 90)*
    %SW70: current time date
    %SW98 - 99: CCOTF compatibility flags for CRA Drop modules
    %SW108: number of currently forced bits
    %SW109: number of forced analog channels
    %SW152 - 155: Ethernet RIO Drop Errors

%SW180 - %SW181: Local drop module health bits (main rack and extension rack)
%SW182 - %SW183: Peer drop module health bits (main rack and extension rack)

NOTE: For more information see %SW180 -%SW183 *(see Unity Pro, Program Languages and Structure, Reference Manual)*.

%SW185 - 339: S908 RIO Drop module health bits
%SW641 - 764: Ethernet RIO Module Health bits

- Reverse System Words:
  %SW62 - 65: data transferred from the Quantum Hot Standby CPU to the Primary CPU

NOTE: For a detailed description of these System Bits and System Words, refer to the Unity Pro Program Languages and Structure Reference Manual *(see Unity Pro, Program Languages and Structure, Reference Manual)*.

User Application Data

- LOCATED:
  All %M, %MW, %MD, %I and %Q data from address 1 up to the maximum number of global address fields configured in Unity Pro's Configuration tab, but no more than 128 KB. A range of %MWs can be defined as a "non-transfer area", they are not transferred to Standby controller.
  - The output (%Q) objects and any output forcing settings.
  - EDT / DDT when they are located by the user.
  - Sequential Function Chart (SFC) data types.
- UNLOCATED:
  - EDT / DDT when they are located by the system.
  - Function Block (EFB / DFB) data types.

The maximum amount of located data that can be transferred in the database is 128 KB for:

- 140 CPU 671 60
- 140 CPU 671 60 S
- 140 CPU 672 61

The maximum amount of unlocated data that can be transferred in the database is for:

- 140 CPU 671 60: 1024 kB
- 140 CPU 672 61: 1536 kB

NOTE: The safety processor, 140 CPU 671 60S, does not use unlocated data.

For specific information about the command words and adjustment parameters and the maximum memory sizes of these areas, refer to the Unity Pro Operating Modes Manual *(see Unity Pro, Operating Modes)*.

For more information on the database transfer, including information about the application of this information by the Standby controller, refer to Quantum Hot Standby Data Transfer *(see page 159)*.

Synchronized Program Execution

By itself, the regular exchange of system and user application data is not enough to synchronize the Standby controller with the Primary controller. It is also important that the cyclical execution of tasks on each controller remains aligned, so that neither controller races ahead of the other controller that is still processing its information. This means that the Primary controller sometimes has to wait for the Standby to finish processing and that the Standby sometimes has to wait for information from the Primary.

This requirement for aligned program execution requires that the task execution cycle is deterministic. For this reason, only MAST tasks are used when programming a Quantum Hot Standby system. For more details about the requirement for MAST tasks and their execution in a Hot Standby context, see Exclusive Use of MAST Tasks *(see page 49)* and Adjusting MAST Task Properties *(see page 168)*.

Switchover Events

While this manual covers Switchover events in some detail, a few general statements aid understanding of these subsequent topics:

- Much of the benefit of the Quantum Hot Standby system is its ability to detect various error conditions and, when necessary, initiate a Switchover. The type of error detected determines the duration of the Switchover event. For example:
  - If the Primary PLC is online and can communicate with the Standby PLC, but detects an error that requires a Switchover, it commands the initiation of a Switchover event. In this instance, the Switchover duration is just that required for the Switchover event, which usually takes about 1.5 - 2 MAST tasks.
  - If the Primary PLC is no longer operable, or all communications between the Primary and Standby controllers are lost, an automatic Switchover occurs. The duration of this type of Switchover equals 2 MAST cycles + any configured Watchdog for the MAST task.
- Local I/O is not part of an automatic Switchover. Local I/O is managed locally (by either the Primary or Standby CPU) and continues to operate after a Switchover through the same controller.

USB Link Switchover Behavior

During a switchover the USB link that is the communication between one of the PLCs and the Unity Pro workstation does not switch over. The link remains with the same PLC, therefore, the link must be manually switched to the other CPU, if necessary.

Quantum Hot Standby Operation Modes

Operating Modes Overview

In a normally operating Quantum Hot Standby system, there are two PLCs running, one as the Primary PLC and one as the Standby PLC. Consequently, a Quantum Hot Standby system requires additional states to reflect the system status. The redundant nature of the system means that the relationships between operating modes changes. The following provides a quick summary of the Quantum Hot Standby operating modes and states.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Verify the PLC operating mode before installing, operating, modifying, or servicing it.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Before acting on a PLC, always positively confirm the operating mode of both Hot PLCs by viewing their LCD displays, LEDs or System Status Words.

Before running a Quantum Hot Standby system that uses an Ethernet RIO Head, ensure that a least one CRA Drop has opened a connection with CRP Head module.

This information is accessible through:
- %SW172 and %SW173
- Mod Status LED of the CRP module, for more information refer to the Quantum Ethernet I/O Ethernet Remote I/O Modules Installation and Configuration Guide

Without a connection opened, the CPU goes to RUN OFFLINE instead of the PRIMARY RUN or STANDBY RUN mode.

For a more detailed description of the Quantum Hot Standby operating modes, including a state transitions diagram, refer to Operating Modes *(see page 123)*.

STOP Mode

In the STOP mode the PLC has both:
- received a Stop command
- successfully stopped

RUN Mode

There are 2 Hot Standby states in the RUN mode:
- Primary state
  The PLC has both:
  - received a RUN command
  - assumed the Primary role because either it did not detect another Primary PLC or, if both PLCs were started simultaneously, it is PLC A
- Standby state
  The PLC has:
  - received a RUN command
  - assumed the Standby role because either it detected a Primary PLC or, if both PLCs were started simultaneously, it is PLC B

NOTE: For information about A/B PLC assignment, refer to Distinguishing Between Controllers *(see page 22)*.

OFFLINE Mode

In the OFFLINE mode the PLC has:
- received a Run command
- responded to a detected error by either changing either from Primary Run or Standby Run to the Offline mode
- received an Offline command

Non-Configured State

In certain circumstances, such as when no valid application is loaded on a PLC, a Hot Standby controller enters (and report itself) as being in a "Non-Conf" (non-configured state), which is not considered as an operating mode.

Remote I/O Management

Overview

Remote I/O is managed only by the Primary CPU, which has all of the RIO features (diagnostics, data exchanges, etc.) available to it.

The CRP Head modules are automatically configured and detect if their CPU is part of a redundant Hot Standby system or Standalone system.

A Quantum Hot Standby system is either:
- S908 system
- Quantum Ethernet I/O Ethernet system

Primary and Standby CRP Head modules report their connection status to their CPUs.

Quantum Ethernet I/O Ethernet CRP Head Modules IP Addresses

A CRP module obtains its IP address assignment at power up as follows:

If the CRP is connected to...	Then the IP address assigned is...
CPU A	IP address configured in Unity Pro for A
CPU B	IP address configured in Unity Pro for B

NOTE: During a Switchover the Quantum Ethernet I/O Ethernet CRPs do not switch IP addresses.

Quantum Ethernet I/O Ethernet CRP Modules and RSTP 2004

Since the CRP A has the lowest priority in a Ethernet daisy loop, therefore, it is the RSPT root. There is only one root in the loop.

CRP B has a higher priority than CRP A, but lower than the CRA RIO Drops and DRSs, therefore, it is the backup root.

If CRP A becomes inoperative, CRP B becomes the root.

However, after a Switchover, if CRP A is still healthy, the root does not change (no loop re-configuration).

If CRP B starts without CRP A in the loop, CRP B becomes the root.

If CRP A starts while CRP B is the root, the loop is re-configured and CPR A becomes the root.

The root CRP reports the status of the Ethernet RIO loop. This information is then transferred to the CPU B during the next scan.

Hot Standby System without RIO

The Quantum Hot Standby system can operate without any Remote I/O installed, but must have linked CRPs installed.

NOTE: This type of S908 Hot Standby system is not compatible with CCOTF.

Drop Hold Up Time

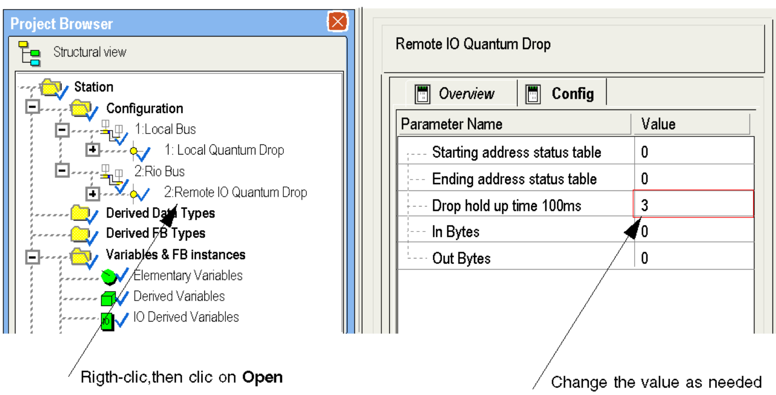
The Drop hold up time must be configured for each Drop module in the Hot Standby system:

- for a S908 system: 1200 ms
- for a Quantum Ethernet I/O Remote I/O Drop: 4 times the MAST task watchdog
- for a Modbus TCP Distributed I/O Drop must be:
  - must be larger than the connection timeout
  - 4 times the MAST task watchdog + 1 second
- for a Quantum Ethernet I/O Distributed I/O Drop: 4 times the MAST task watchdog + 1 second

NOTE: Schneider-Electric devices can have different configurations of drop holdup time and connection timeout, but third party devices cannot. They are not compatible with a Quantum Hot Standby system.

How to Configure Hold Up Time Value

The following table describes the procedure to change the hold up time value:

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>Create a RIO bus with a 140 CPU 67• 60 Quantum processor and a 140 CRP •••_00 Head module.</td></tr>
<tr><td>2</td><td>Add a rack on RIO bus with a 140 CRP •••_00 Head module.</td></tr>
<tr><td>3</td><td>Open the Remote IO Quantum Drop dialog and change the drop hold up time.

</td></tr>
</table>

Hot Standby Programming Differences

Overview

In general, programming a Quantum Hot Standby controller with Unity Pro is very similar to programming any other standalone Quantum controller using Unity Pro. Unity Pro provides a user-friendly, IEC 61131-3 compatible development environment. Most of your programming skills in other development environments and for other devices are applicable for the Quantum Hot Standby.

However, there are some important considerations:

- The application programs on both PLCs must be identical. If not, the PLC reports a “logic mismatch":
  - If the Hot Standby PLCs are operational at the time a logic mismatch occurs, the Standby controller enters the Offline operating mode.
  - If a logic mismatch exists during a simultaneous startup of both the Hot Standby PLCs, one PLC starts as the Primary and the other PLC remains in the Offline operating mode.
  - If the controllers are started sequentially and a logic mismatch exists, the second PLC that attempts to start starts in the Offline mode.
  - When the Hot Standby controllers test for a logic mismatch, they check if the application program loaded on both PLCs are identical.
  - If the application programs on each PLC are different, the result is a logic mismatch.
  - Some changes to the application programs are possible while online; other changes require an offline update. For more information, refer to Application Modifications *(see page 186)*.

- When connecting Unity Pro to a Hot Standby system, keep in mind that:
  - Generally, the information in Unity Pro is the same whether you connect to the Primary PLC or to the Standby PLC. Most registers on the Standby PLC reflect the values provided by the Primary PLC during each MAST task.
  - Some differences between the data on the Primary PLC and the Standby PLC exist. These exceptions include the located System Word (%SW61) and User Application data maintained independently on each PLC.
  - Writing values to the Standby PLC registers is ineffective because the next database transfer from the Primary PLC overwrites these values.

NOTE: Only unlocated data in the non-transferred area are not overwritten by data from the Primary PLC.

Application Task Types

In a Quantum Hot Standby system, the Standby controller must remain ready to assume the role of the Primary controller. This requires that both controllers run identical applications, and that the Standby controller is provided with current application data and state information from the Primary controller once per scan. The synchronous and deterministic transfer of the Primary controller data and state information to the Standby controller is achieved by using MAST tasks.

Exclusive Use of MAST Tasks

Only the MAST tasks must be used in a Quantum Hot Standby system because the transfer of Primary PLC system and user application data to the Standby PLC is synchronized in each MAST task cycle, refer to Second Step of Execution Time Measurement *(see page 170)*.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Do not use programming methods based on data that are not synchronized in each MAST task cycle.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

The following programming methods are examples that must not be used in a Hot Standby application:
- preemptive, asynchronous, or interrupt-driven (EVENT) tasks
- FAST tasks
- intermediate I/O
- section schedulers
- events and edge triggers, etc.
- IU_ERIO function block

They can impact the performance of the MAST tasks and cause discrepancies between Primary and Standby output values in the event of a Switchover.

Be careful when using explicit messages and time stamping:
- If explicit messages are used, some messages may be sent twice and answers may be lost during a switchover.
- If time stamping are used, some time stamps may be lost during a switchover.

Only MAST tasks support data synchronization between the Primary and Standby controllers.

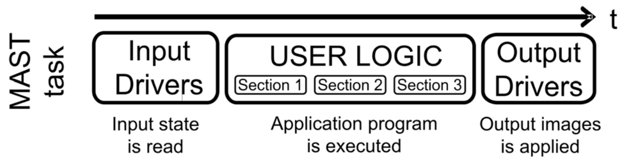
How Hot Standby MAST Tasks Differ

Hot Standby MAST tasks are different from the normal MAST tasks from Standalone Quantum PLCs. In a Quantum Hot Standby PLC, the execution of a MAST task involves extra steps necessary to support redundancy.

These additional steps provide the following:
- Database transmission.
- Wait states to synchronize MAST task execution (see Synchronized Program Execution *(see page 43)*) between the two PLCs.

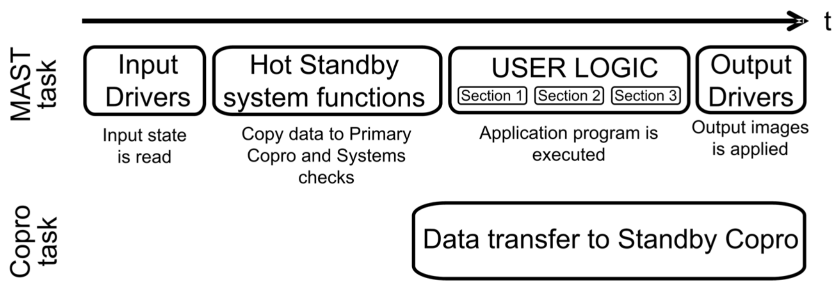
MAST Tasks Comparison

An example of a Standalone MAST task follows:



The Hot Standby version of the MAST task introduces an additional step for the “Hot Standby System Functions”, the transmission of the database from the CPU to the Copro.

An example of a Hot Standby version of the MAST task follows:



The time required to transfer the database to the Copro, and for the Copro to communicate this information to the Standby, scales linearly with the size of the database. For more information on Hot Standby MAST tasks actions and durations, refer to Database Exchange *(see page 41)* and Adjusting MAST Task Properties *(see page 168)*.

Debugging

Debugging your Hot Standby application program is now a two-stage process:

1. The application on a single Hot Standby PLC as if it were a standalone application. This allows the use all of the debugging features available in Unity Pro, such as watch-points, etc.
2. Debug your application when it has been uploaded to two Hot Standby PLCs in a working redundant system, but in a non-production environment. On this platform, evaluate performance specific to Hot Standby redundancy. Only a subset of Unity Pro's debug features can be used during this stage.

NOTE: See Debugging Your Hot Standby Application *(see page 175)* for further details.

Primary, Standby or Offline Execution

In a Quantum Hot Standby system, your application is executed differently depending on whether it is running on the Primary PLC or on the Standby PLC. The full application program is executed on the Primary controller, while the Standby only runs the first section of the MAST task.

According to the user configuration, the Offline PLC can execute:

- full program
- only the first section of the MAST tack
- none of the MAST program task

This is important because some system behaviors must be commanded in the first section of the MAST tack. For example, the Standby PLC's Reverse Transfer Registers (%SW62 -%SW65) can contain custom diagnostic information for use by the full program on the Primary PLC.

1.2 Hot Standby Safety CPUs

Overview

This section describes the use of the Quantum Safety CPU, 140 CPU 671 60S, in a Quantum Hot Standby system.

NOTE: This CPU cannot be used in a Quantum Ethernet I/O Hot Standby system.

What's in this Section?

This section contains the following topics:

Hot Standby Safety CPU Specifics

Introduction

The 140 CPU 671 60S Quantum Safety CPU module is certified for use in Hot Standby SIL3 solutions compliant with the 61508 IEC standard. For more details about to the safety certifications, see the Quantum Safety PLC *(see Modicon Quantum, Quantum Safety PLC, Safety Reference Manual)*.

In the Standalone Safety CPU, the Ethernet port is used to communicate with other devices using a normal Ethernet cable.

In the Hot Standby Safety CPU, the connection used to exchange data between the Primary CPU and the Standby CPU controller is a fiber optic link. Because the fiber optic link is not part of the Safety loop, the PFD and PFH values of the Hot Standby CPU are the same as those of the Standalone CPU.

Each Safety CPU can include a PCMCIA memory card *(see Modicon Quantum, Quantum Safety PLC, Safety Reference Manual)*, but its use and presence is not mandatory.

NOTE: This CPU cannot be used in a Quantum Ethernet I/O Hot Standby system.

Description of a Safety Hot Standby Configuration

The Hot Standby configuration contains two identical local racks and at least one remote I/O drop because I/Os cannot be placed in the local rack of a Safety Hot Standby configuration.

Besides a power supply module (there must be at least one 140 CPS 124 20), each local rack must contain:
- 140 CPU 671 60S module
- 140 CRP 932 00 module

Besides a power supply, I/O modules (including at least one 140 CPS 124 20), the remote drop(s) must include a 140 CRA 932 00 module.

⚠ CAUTION

UNINTENDED EQUIPMENT OPERATION

Use only high availability RIO modules with dual cabling in a Safety-Related System.

Failure to follow these instructions can result in injury or equipment damage.

Description of the Operating Modes

- **Safety Mode:** This is the default mode. It is a restricted mode in which modifications and maintenance activities are prohibited.
- **Maintenance Mode:** This is a temporary mode for modifying the project, debugging and maintaining the application program.

State Compatibility with Safe and Maintenance Modes

A Quantum Hot Standby system has two states:

- **Redundant (1 CPU is Primary, 1 is Standby)**
  The Standby CPU controller mode follows the Primary CPU controller mode. For example, if you switch the Primary CPU controller from Safety to Maintenance mode, the Standby CPU controller switches from Safety to Maintenance mode at the start of the next cycle.
- **Non-redundant (at least 1 CPU Offline)**
  The two controllers are independent, one can be in Safety mode and the other one in Maintenance mode. For example, the Run Primary controller can be in Safety mode while the Stop Offline controller is in the Maintenance mode.

Impact of the PLC Switchover on the Process Safety Time

If the Primary CPU detects an internal or external problem, it stops exchanging data with the Standby CPU and stops processing the I/O. As soon as the Standby CPU detects that there are no longer exchanges with the Primary CPU, it takes over the role of the Primary CPU, executing the user logic and processing the I/O. Therefore, the output modules must filter the lack of exchange with the Primary CPU to avoid glitches when a Switchover occurs. This is achieved by configuring the output module timeout. As a result, the PLC reaction time is greater than the timeout configured in the output module, thereby influencing the process Safety time.

NOTE: The behavior of the Hot Standby Safety CPU is equivalent to a Standalone Safety CPU.

In case of a detected error, the Safety PLC enters:

- Halt state when running in the Maintenance Mode
- Error state when running in the Safety Mode

Availability of the Hot Standby Functions

In addition to the standard Hot Standby functions, you can use an EFB to program an automatic switchover between Primary CPU and Standby CPU to verify the ability of the Standby CPU to take over from the Primary CPU. That means that the Standby CPU periodically becomes the Primary CPU and the Primary CPU becomes the Standby CPU.

It is recommended to avoid using the USB link during a Switchover.

The following table lists the available Hot Standby functions in Maintenance and Safety modes:

Function	**Maintenance Mode**	**Safety Mode**
Hot Standby	yes	yes
Switchover	yes	yes
EFB Swap	no	yes
Keypad	yes	yes
Application mismatch	yes	no
OS Upgrade	yes, if Standby CPU is in Stop Offline	no
Application Transfer	yes	no

Operating Modes of the Safety PLC

Introduction

The default behavior of the Quantum Safety PLC is to perform Safety Functions to achieve and to maintain the Safe state of a process. Nevertheless, you must be able to debug and to maintain your project.

Use the Safety Mode to control your process and the Maintenance Mode for debugging and refining your project.

In Maintenance Mode, the I/O and CPU modules are still executing the diagnostics and establishing the Safe state if a fault is detected. Only the application program and the application data, which may be changed in Maintenance Mode, are not checked.

NOTE: To program a Safety PLC, Unity Pro XLS is required.

Safety and Maintenance Mode Features

The operating mode of the Quantum Safety PLC depends on events such as application exception, power on/off, and so on. The functions available in Unity Pro XLS depend on the operating mode.

Switching between the modes requires defined conditions and follows certain procedures. For details, see the chapter "Switching Between Safety and Maintenance Mode" *(see Unity Pro XLS Software, Operating Mode Manual, Safety PLC Specifics)* in the *Unity Pro XLS Operating Mode Manual Safety PLC Specifics*.

You can interact with the Safety PLC using:
- Unity Pro XLS programming tool
- Quantum Safety CPU keypad
- Quantum Safety CPU key switch

Depending on the operating mode, the Safety PLC can be in different states.

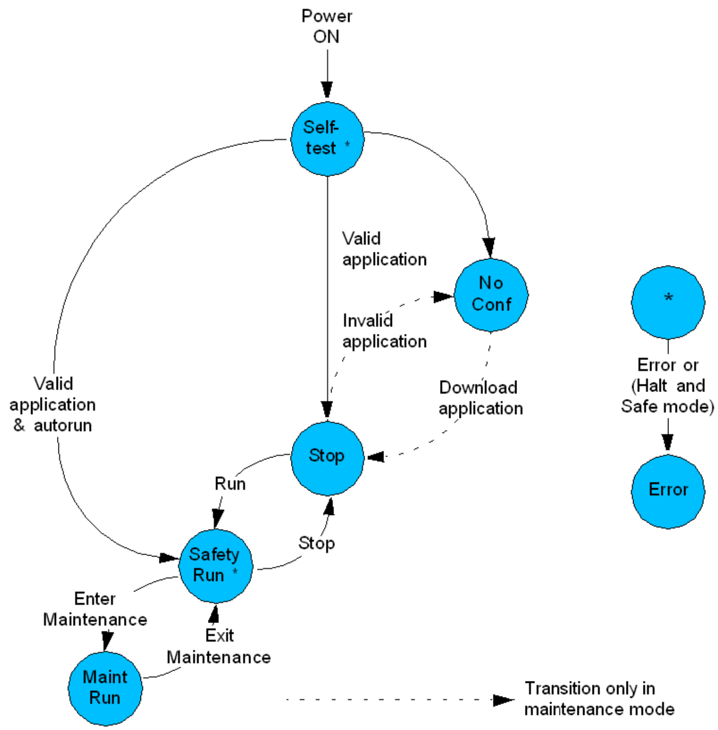
After power up, it automatically enters run state of the Safety Mode if the following 2 conditions are fulfilled:
- There is a valid application.
- The **Automatic start in Run** option is activated.

In case of an invalid application, it enters the not configured (no conf) state of the Maintenance Mode (only if the key state is unlocked), in which you are able to download your project.

If a fault is detected, the PLC enters
- Halt state when running in Maintenance Mode.
- Error state when running in Safety Mode.

PLC States

The following figure shows the state diagram of the Quantum Safety PLC:



Operating Mode Identification

The LCD display on the CPU indicates the current operating mode by showing the letters *M* for Maintenance Mode or *S* for Safety Mode.

The status bar field on the PLC screen indicates the current operating mode as shown in the following figure:

Configuring and Maintaining a Quantum Hot Standby System

II

Overview

This part describes three important processes in using a Modicon Quantum Hot Standby system:
- configuring a Quantum Hot Standby system using the Unity Pro software
- installing and cabling a Quantum Hot Standby system
- maintaining a Quantum Hot Standby system once installed

What's in this Part?

This part contains the following chapters:

Configuring with Unity Pro

2

Overview

This chapter provides an overview of using Unity Pro to configure registers and program a Quantum Hot Standby system.

What's in this Chapter?

This chapter contains the following sections:

2.1 Unity Pro Tabs and Dialogs

Purpose

Use the Unity Pro editor dialog tabs to:
- Select options for configuring the Quantum Hot Standby 140 CPU ••• 60(60S)
- Obtain system status information

What's in this Section?

This section contains the following topics:

Introducing Unity Pro

Overview

Unity Pro software is a fully Windows compatible application. Unity Pro supports only the IEC methods of configuration.

No Loadables Needed

Unlike legacy Modicon Quantum where the CHS module owns the control functionality, the Unity Pro Modicon Quantum Hot Standby with Unity systems has the control functionality embedded in the Executive.

Command Register

The Command Register defines the basic operational parameters of a Modicon Quantum Hot Standby with Unity solution. The command register's functionality is described in *Hot Standby Command Register, page 86*.

Opening the Editor Dialog

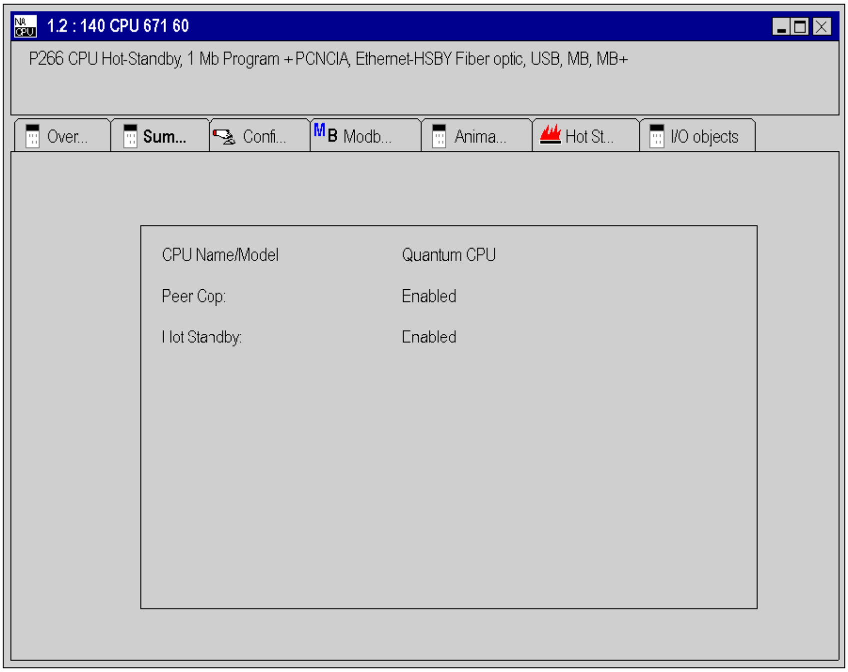
After starting Unity Pro, go to the Local Bus in the Structural View of the Project Browser.

Step	Action
1	Open the Local configuration editor either by double-clicking on the Local Bus or by selecting the Local Bus and executing right-click Open A graphical representation of the local bus appears in the configuration editor.
2	Select the Modicon Quantum Hot Standby with Unity 140 CPU 671 60/60S module and right-click. The context menu appears.
3	Select Open Module.
4	The editor appears. The Summary tab is the default.

Using the Summary Tab

Viewing

Use the Summary tab of the Unity Pro editor to determine if Peer Cop and Hot Standby are enabled.



Describing

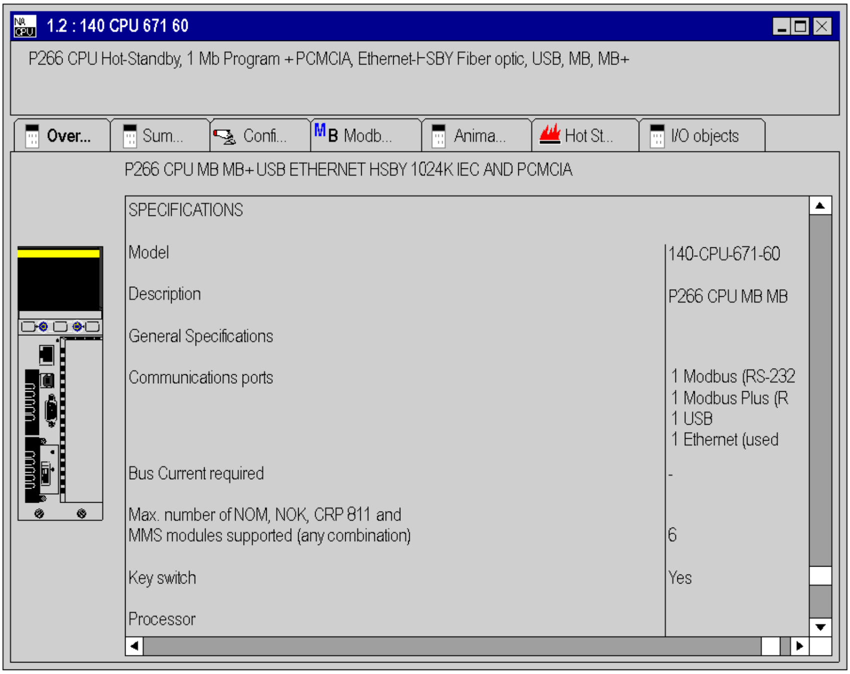
Summary tab:

Item	**Option**	**Value**	**Description**
CPU Name/Model:	Quantum CPU	N/A	Read Only
Peer Cop:	Disabled	Enabled	Read Only
			Peer Cop="Enabled" if the function is valid in the Modbus Plus menu
Hot Standby:	Enabled	Enabled	Read Only

Using the Overview Tab

Viewing

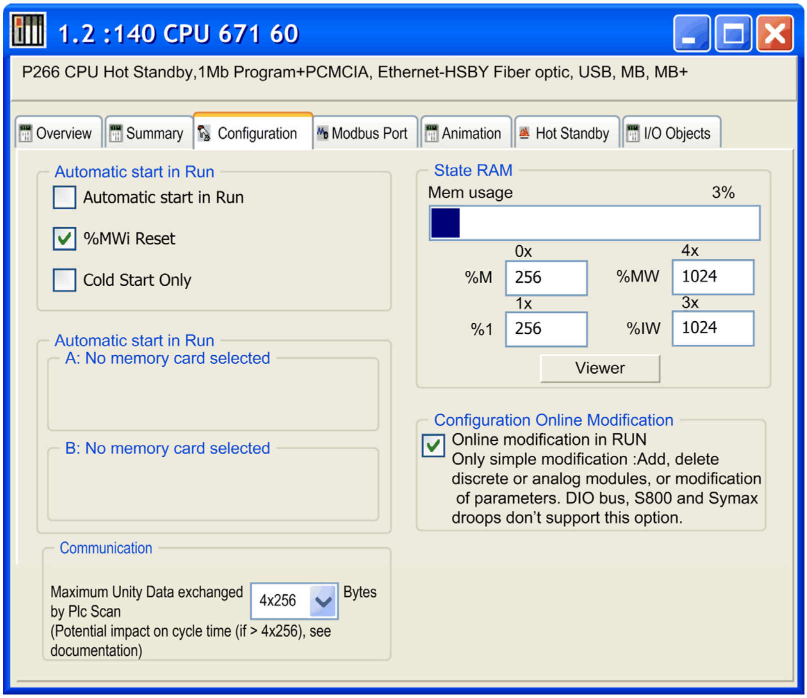
The read only Overview tab of the editor displays detailed information about the module’s specifications.



Using the Configuration Tab

Configuration Screen

Change values using the Configuration tab of the editor.



Description

Configuration tab:

Item	Option	Value	Description
Operating Mode On Cold Start	Automatic start in Run	x	Determines the operating condition during Cold Start
	`%MWi` Reset on cold start	x	
	Cold Start Only	x	Enables the Cold Start Only *(see page 69)* feature.
Memory Cards	A:	N/A	Displays the configuration in the PCMCIA Slots
	B:	N/A	

Item	Option	Value	Description
Communication	By default, the bandwidth is 4x256 bytes, supported by the OS versions prior to V2.80 for the CPU and V4.60 for the NOE.		The maximum data volume exchanged each cycle between the NOE and CPU modules
	For Quantum processors: • 140 CPU 311 10 • 140 CPU 534 14 • 140 CPU 434 12	4x256 4x1024	
	For Quantum processors: • 140 CPU 651 50 • 140 CPU 651 60 • 140 CPU 652 60 • 140 CPU 671 60 • 140 CPU 672 61	4x256 4x1024 8x1024 12x1024	
State RAM	Mem usage	1.	A bar displays percent of memory used.
	`%M`-0x	2.	Size of the different memory areas **Note:** The values for `%IW` and `%MW` have to be divisible by 8.
	`%MW`-4x	2.	
	`%I`-1x	2.	
	`%IW`-3x	2.	
	Viewer	N/A	Opens the State RAM Viewer tab, which displays the allocation of used memory. (See the illustration following.)
Configuration Online Modification	Online modif in RUN	x	This check box allows to: • Add or delete discrete or analog modules, • Modify Parameters **NOTE:** These modifications can be done in RUN mode.

1. The value (expressed as a percentage and displayed on the scale) depends on the memory usage of the Hot Standby configuration.
2. Enter the appropriate values. All values depend on Hot Standby configuration.

Automatic Start in RUN

The enabling of this option automatically changes the PLC to Run mode *(see Unity Pro, Program Languages and Structure, Reference Manual)* on cold start.

Two types of start:
- in the absence of a PCMCIA memory card, the PLC starts on the contents of the internal RAM of the processor
- in the presence of a PCMCIA memory card it is its content which fixes the start

WARNING

UNWANTED APPLICATION RUN ON PLC COLD START

With the **Automatic start in RUN** option enabled, the following events will trigger the run of the application on cold start:
- Inserting the PCMCIA card when the PLC is powered
- Replacing the processor while powered
- Unintentional or careless use of the reset button
- Powering up a PLC with a defective battery after a power outage

To prevent the run of the application on cold start:
- use the STOP input (on Premium PLCs)
- use the switch on the front panel of the processor (for Quantum PLCs)

Failure to follow these instructions can result in death, serious injury, or equipment damage.

%MWi Reset

On application download:
- if you check the box, the %MWi values will be re initialized or set to 0
- if you uncheck the box, the %MWi values will set to 0

On cold start or after inserting the PCMCIA memory card:
- if you check the box, the %MWi values will be re initialized or set to 0
- if you uncheck the box, the %MWi values will retain their current value

Cold Start Only

If checked, this option forces the cold start *(see Unity Pro, Program Languages and Structure, Reference Manual)* of the application, instead of the normal warm start *(see Unity Pro, Program Languages and Structure, Reference Manual)*.

By default, the Cold Start Only option is unchecked.

The Cold Start Only option is only supported on High End PLC since V2.7.

An application using this functionality will not be:
- downloadable on a PLC with a previous version
- executable on a PLC with a previous version
- usable with Unity Pro V4.0 or lower

NOTE: The Cold Start Only check box is present only if the current selected PLC can support it.

Communication

When the **UNITY** protocol under TCP/IP is used (OFS or Unity Pro), it is possible to configure the maximum volume of data that can be exchanged each cycle between the CPU and the NOE modules using the **Maximum** Unity Data exchanged by Plc Scan **option**.

This functionality is only supported on CPU modules with OS version 2.80 or higher, and on NOE modules with OS version 4.60 or higher.

The bandwidth set is valid between the CPU and all existing NOE modules. It is not possible to set different bandwidths for each of the modules.

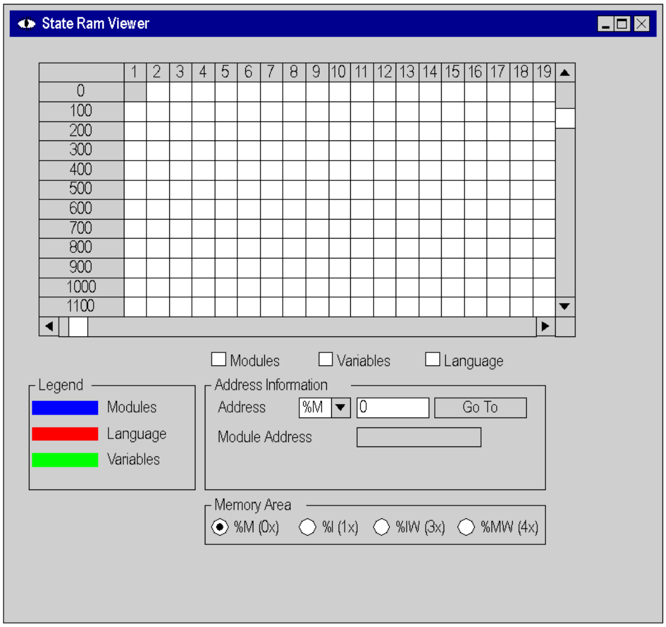
Increasing this bandwidth has an impact on the cycle time of the controller (2 ms per kbytes exchanged). This impact is proportional to the amount of data actually exchanged and not the bandwidth configured. So, if the channel is set to the maximum, but not used, the impact on cycle time will be negligible.

State RAM memory

The **State RAM** bar chart allows you to know the size of the **State RAM** memory used in your project in relation to the maximum memory size.

Using the State RAM Viewer

The State RAM Viewer dialog:



Each cell in the grid represents an address location and displays the entity stored in that location. The contents of the grid may be changed by selecting options in either of two filters:

1. Memory used grid options:

Select one—or all—of the three options (using the check box) and one to three bar graphs appear.

- Modules
  Indicates the topological address used in the modules. Address appears as a bar graph in the grid.
- Language
  Indicates the topological address used in the program. Address appears as a bar graph in the grid.
- Variables
  Indicates the topological address used in the variables. Address appears as a bar graph.

2. Memory Area options:
Using this option, you designate a state RAM address. Select one of four reference types.

- `%M`
- `%I`
- `%IW`
- `%MW`

Your choice appears in the Address field of the Address Information area.

Online Configuration Modification

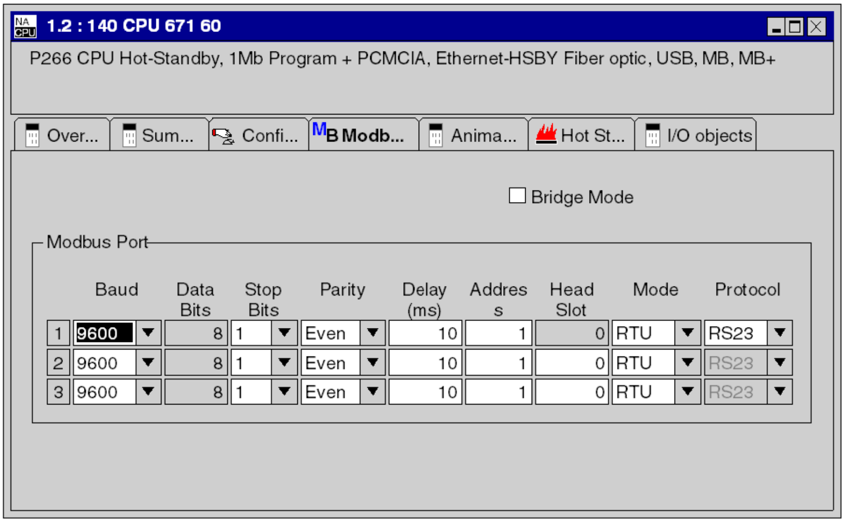
On PLCs supporting this functionality a check box is activated and appears in the CPU Editor *(see page 66)*.

The Configuration Online modification is only available on certain types of PLCs *(see Unity Pro, Operating Modes)* if the **online modif in RUN** check box is selected.

Using the Modbus Port Tab

Viewing

You may change Modbus communication options using the Modbus Port tab of the Unity Pro editor:



NOTE: If you need the Modbus address of the controller, go to the 140 CPU 671 60 module and find the address using the keypad *(see page 225)*.

Configuring Modbus Plus (MB+) Addresses

When configuring the MB+ address for the first time:

- Default MB+ address = 1 (140 CPU 671 60/60S)
- Change MB+ address at first configuration (on both controllers) *(see page 62)*

WARNING
UNINTENDED EQUIPMENT OPERATION. Do not change the Modbus Plus (MB+) address after the first configuration. **Failure to follow these instructions can result in death, serious injury, or equipment damage.**

Describing

Modbus Port tab:

Item	Option	Value	Description
Modbus Port	Baud	9600	Data must be specified for every link.
		50-19200 kBit/s	
	Data Bits	8	
	Stop Bits	1 or 2	
	Parity	EVEN	
		ODD	
		NONE	
	Delay (ms)	10 ms	
	Address	1 -247	
		For Modbus switchover 1 - 119 (Primary CPU) 129 - 247 (Standby CPU)	
	Head Slot	0	
	Mode	RTU	
		ASCII	
	Protocol	RS232	
		RS485	

Using the Animation Tab and PLC Screen Dialogs

Accessing the PLC Screen Dialogs

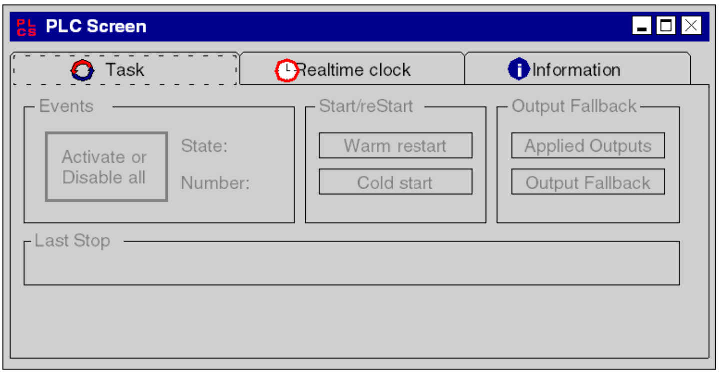
To access the Task, Realtime clock, and Information tabs of the Unity Pro Animation tab:

Step	Action
1	Select the Animation tab.
2	The PLC screen tab appears automatically.

NOTE: The dialogs illustrated here are depicted in offline mode. When Unity Pro is connected to a PLC, the information displayed in these tabs changes.

Viewing the Task Tab

Unity Pro Task tab dialog:



NOTE: Click to see the PLC screen in online mode and the corresponding description *(see Unity Pro, Operating Modes)*.

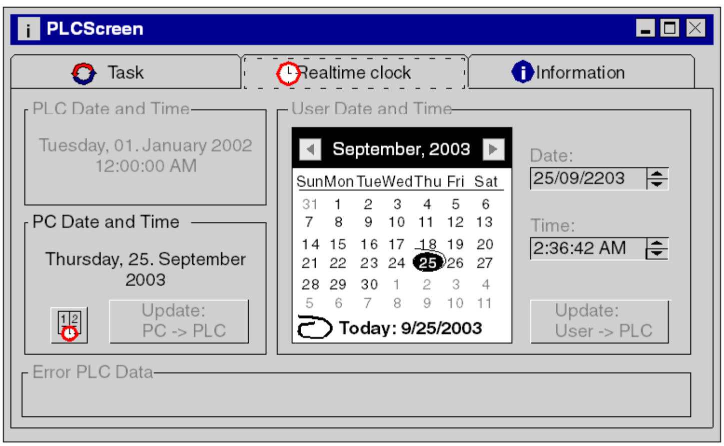
Describing the Task Tab

Description of the Task tab:

Item	Option	Value	Description
Events	State:	xxx	Status information of events available Online
	Number:	xxx	N/A
	Activate or Disable all	Click button	Button to control the events
Start/reStart	Warm Start	Click button	To initialize Warm Start
	Cold Start	Click button	To initialize Cold Start
Output fallback	Applied Outputs	N/A	Not used in Modicon Quantum Hot Standby with Unity system
	Output Fallback	N/A	
Last Stop	Read only	• Day • DD/MM/YY • Time	Indicates the day, date, time, and cause of the last controller stop

Viewing the Realtime Clock Tab

Unity Pro Realtime clock tab dialog:



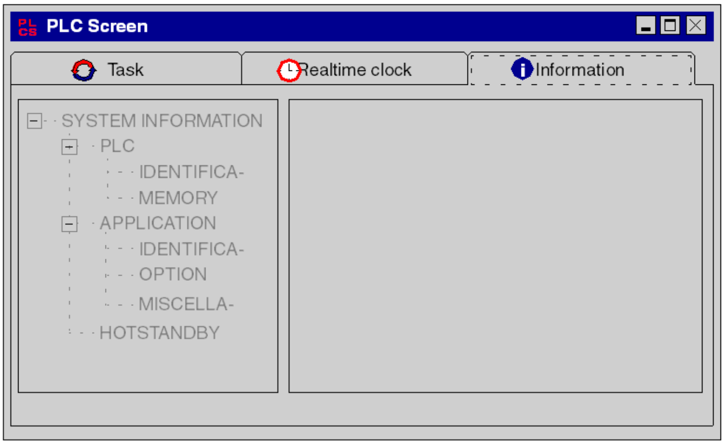
Describing the Realtime Clock Tab

Description of the Realtime clock tab:

Item	Option	Description
PLC Date and Time	Read only	Indicates the current PLC date and time
PC Date and Time	Update PC->PLC	Updates the PLC with the PC system time
User Date and Time	Update User->PLC	Updates the PLC with the time set by the user

Viewing the Information Tab

Unity Pro Information tab dialog:



Describing the Information Tab

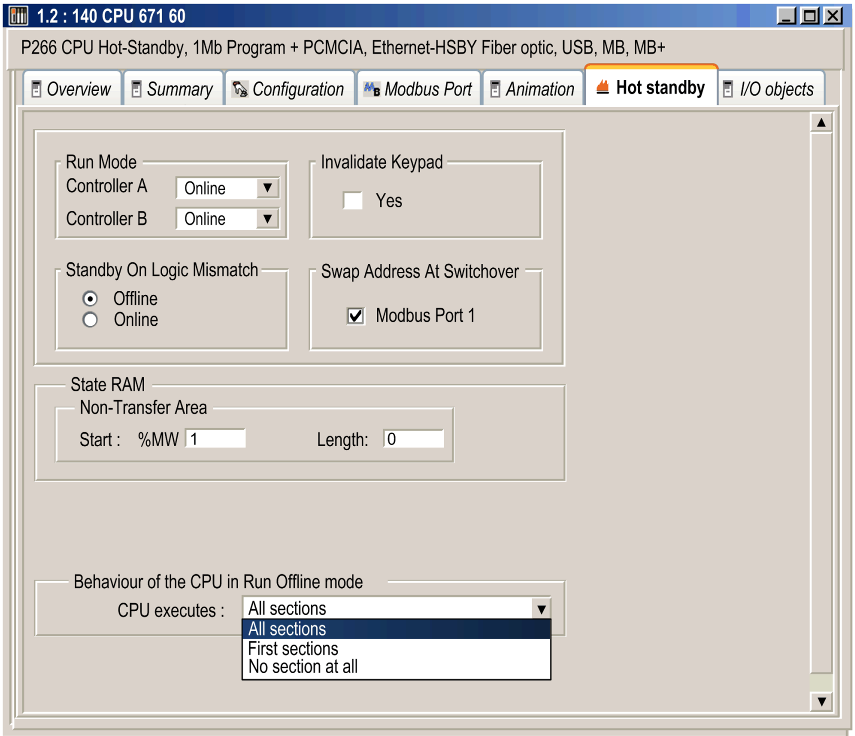
Description of the Information tab:

Item	Option	Value	Description
System Information	PLC / Identification	PLC Range	Only Online available
		Processor Name	
		Processor Version	
		Hardware ID	
		Network address	
	PLC / Memory	RAM CPU	
	Application / Identification	Name	
		Creation Product	
		Date	
		Modification Product	
		Date	
		Version	
		Signature	
	Application / Option	Upload Information	
		Comments	
		Animation Table	
		Section Protection	
		Application Diagnostic	
	Application / Miscellaneous	Forced Bits	
	Hot Standby	PLC Hot Standby Status	
		Peer PLC Hot Standby Status	
		Application mismatch between PLC and Peer PLC	
		PLC Name	
		Variable Transfer Status	
		Hot Standby Entire System State	

Using the Hot Standby Tab

Viewing the Hot Standby Tab

Configure Hot Standby values in the Hot Standby tab of the Unity Pro editor:



Description of the Hot Standby tab:

Item	Option	Value	Description
Run Mode	Controller A	Offline/Online	Indicates which controller is Offline and Online at the next start up.
	Controller B	Offline/Online	
Invalidate Keypad	Disable	Yes is NOT selected	When selected, the keypad cannot be used to change the Hot Standby submenu.
	Enable	Yes is selected (Check mark displayed)	

Item	Option	Value	Description
Standby On application mismatch	Offline	Default Offline button selected	If mismatch is detected, Standby goes Offline
	Online	Default Online button not selected	If button is selected and mismatch is detected, Standby remains Standby
Swap Address At Switchover	Modbus Port 1	Default All selected	When selected, enables Modbus switchover to occur.
State RAM: Non-Transfer Area	Start: `%MW`	1	Starting address of memory area not transferred.
	Length	1	Specify the range of the length.
Behavior of the CPU in Run Offline mode	All sections of the MAST task	Default	Regarding the option selected, the CPU will or will not execute the program when CPU is in Run Offline mode.
	First section of the MAST task		
	No section of the MAST task		
1. Enter the appropriate values. All values depend on Hot Standby configuration.			

Local Drop Module Diagnostics (Status Table)

In a Quantum Hot Standby system the CPUs exchange Drop diagnostic WORDs.

In a Hot Standby system the number of WORDs exchanged is increased to 32 from the Standalone 16 WORDs.

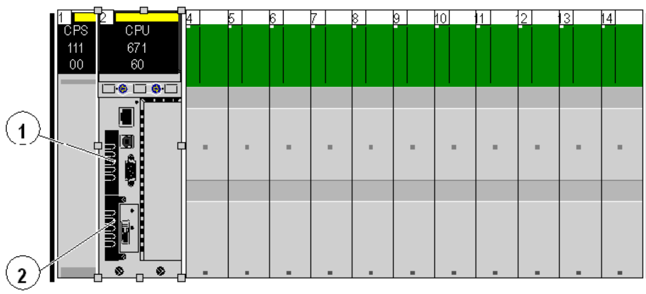
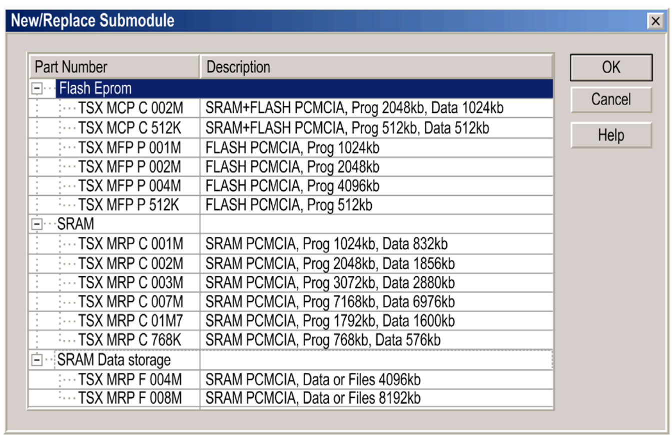
The first 16 WORDs contain the local CPU diagnostics.

The second 16 WORDs are used to exchange the local diagnostics between the 2 CPUs.

Configuring the PCMCIA Cards

Configuring with Unity Pro

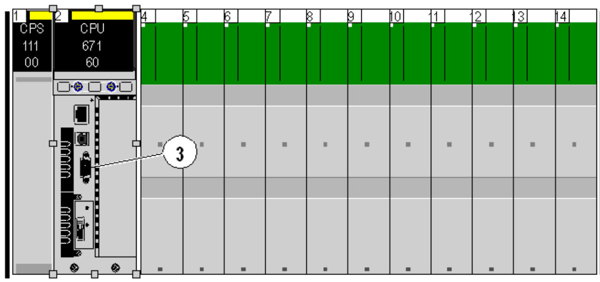
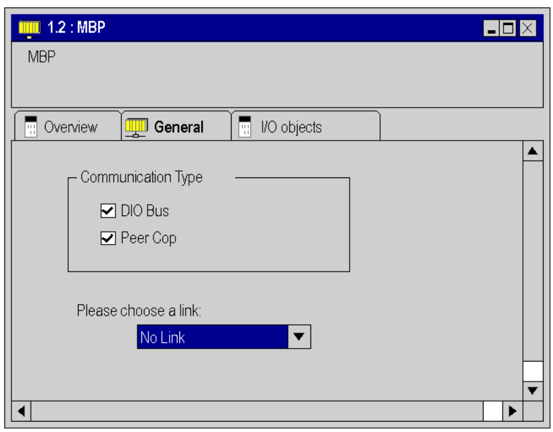
This is the procedure to allocate memory to the memory card:

Step	Action
1	If not opened, open the Local Bus configuration editor.
2	Go to the local bus in the Structural View of the Project Browser.
3	Open the local bus either by double-clicking on the Local Bus or by selecting the Local Bus and executing right-click Open. **Result**: A graphical representation of the local bus appears.
4	Select either PC Card A (1 slot) or PC Card B (2 slot).  **1** Memory configuration of the PCMCIA card 1 **2** Memory configuration of the PCMCIA card 2
5	Double-click or right-click either PCMCIA card. **Results**:The New/Replace Submodule dialog appears. 
6	Add or replace the desired memory.

Configuring the Modbus Plus Communication Type

Configuring with Unity Pro

This procedure configures the Modbus Plus communication type

Step	Action
1	If not opened, open the Local Bus configuration editor.
2	Go to the local bus in the Structural View of the Project Browser.
3	Open the Local Bus editor either by double-clicking on the Local Bus or by selecting the Local Bus and executing right-click Open. **Results**: A graphical representation of the local bus appears.
4	Point to the Modbus Plus port, No. 3. 
5	Double-click or right-click on the Modbus Plus port. **Results** The Submodule dialog appears. The General tab is the default. 
6	Select one or both Communication Type: ● DIO bus ● Peer Cop

Non-Transfer Area and Reverse Transfer Words

Non-Transfer Area of State RAM

The designated registers in the Non-Transfer Area is ignored when state RAM values are transferred from the Primary CPU controller to the Standby CPU. Placing local date and I/O registers in the Non-Transfer Area can reduce the scan time.

NOTE: Due to the hardware design of the Quantum Hot Standby CPU processors, the scan time optimization provided by the Non-Transfer Area may be very low.

Using the Hot Standby tab of the editor dialog, designate a block of %MW words as a Non-Transfer area:

STEP	Action
1	Ensure that the Hot Standby tab is selected. If you want to review the process for starting Unity Pro and opening the editor dialog, please see Configuring Unity Pro Dialogs *(see Quantum with Unity Pro, Hardware, Reference Manual)*.
2	Enter the starting address in the system word `%MW` field. The field is located in the Non-Transfer Area of the Hot Standby tab.
3	Enter the number of contiguous registers in the **Length:** field. The field is located in the Non-Transfer Area of the Hot Standby tab.

Transferring Standby CPU Data to the Primary CPU

The system words,%SW62/63/64/65 are dedicated to transfer data from the Standby CPU controller to the Primary CPU.

These system words can be used by the application program (in the first section of the MAST task) to register diagnostic information.

The data coming from the Standby CPU are transferred at each scan and are available to the Primary CPU.

When the secondary CPU is Offline, the reverse registers are not transferred to the Primary. If the user does not change the value on Primary side, this previous value is kept. When Secondary becomes the Standby, the reverse registers are updated on the Primary side 2 MAST cycles after the transition.

Setting Up the Quantum Hot Standby System

Overview

Setting up a Quantum Hot Standby system involves a number of processes, summarized in the following paragraphs here and explained in detail elsewhere.

Mapping the Rack Extensions

A Quantum Hot Standby requires two racks with at least four slots. Map the two racks in an identical manner as described in Identical Configurations *(see page 39)*.

Connecting Two CPUs

Connect the two Quantum Hot Standby CPUs with a fiber optic cable as described in Hot Standby Sync-Link Topology *(see page 27)*.

Establishing the Primary CPU and Standby CPU Controllers

The system determines that one of the two Quantum Hot Standby CPUs is the Primary, the A, CPU (and that the second CPU is the Standby, the B, CPU), refer to Establishing the Primary and Standby Controllers *(see page 23)* and Distinguishing between Controllers *(see page 22)*.

The Keypad may provide status information. Therefore, to view the status, use the Quantum Hot Standby CPU keypad by selecting **Quantum PLC Operations →PLC Operations Hot Standby →Hot Standby Order**.

Refer to CPU Controls and Displays *(see page 220)* and Using the CPU LCD Display Screens *(see page 224)*.

Configuring in Unity Pro

Using Unity Pro, configure a network that is appropriate for the installed racks and the cabling.

Configure the Hot Standby Register for the Quantum Hot Standby CPU in Unity Pro as described in Configuring the Unity Pro Dialogs *(see Quantum with Unity Pro, Hardware, Reference Manual)*.

Transferring and Sending the Program from Primary CPU to Standby CPU

Transfer the program from your PC to CPU using the Unity Pro command **PLC → Transfer program to PLC**.

Refer to Application Program Transfer *(see page 160)*.

Send your program from the Primary CPU to the Standby CPU using the Primary or Standby CPU keypad. Select **Quantum PLC Operations →PLC Operations Hot Standby →Hot Standby Transfer →Press <ENTER> to confirm Transfer**.

Refer to Using the HE CPU 67160 LCD Display Screens *(see page 220)*.

NOTE: A program always goes from the Primary CPU controller to the other CPU controller.

2.2 Reading and Configuring Registers

Purpose

This section describes configuring the command register of a Quantum Hot Standby system by selecting options that affect the register. You may want to use this method if your system has specific configuration needs.

This section also describes the read-only status registers.

What's in this Section?

This section contains the following topics:

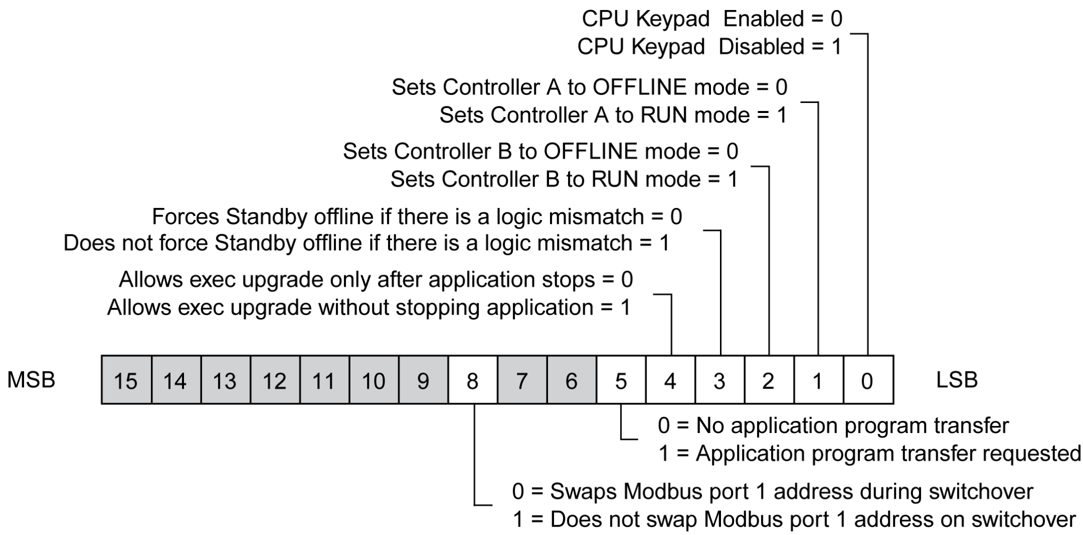
Hot Standby Command Register

Setting the Bits in the Command Register

The Command Register (System Word), `%SW60`, defines the operating parameters of a Hot Standby application for both the Primary CPU and Standby CPU.

At each scan, the Command Register is replicated and transferred from the Primary CPU to the Standby CPU. Transfer occurs only from Primary CPU to Standby CPU. Any changes made to the Command Register on the Standby CPU has no effect because the values transferred from the Primary CPU overwrite the values in the Standby CPU.

The following illustration identifies the operating options provided by the Command Register:



System Bit %SW60.0

Invalidate Keypad is an option that allows a controller to refuse commands from the Hot Standby submenu in the front panel keypad.

- `%SW60.0 = 1`
  Invalidate Keypad enabled.
  The Quantum Hot Standby system refuses all changes from the Hot Standby submenu in the front panel keypad.
- `%SW60.0 = 0`
  Invalidate Keypad disabled.
  The Quantum Hot Standby system accepts all changes from the Hot Standby submenu in the front panel keypad.

System Bit %SW60.1

Controller A OFFLINE/ONLINE mode:

- `%SW60.1 = 1`
  Controller A goes to the ONLINE mode.
- `%SW60.1 = 0`
  Controller A goes to the OFFLINE mode.

System Bit %SW60.2

Controller B OFFLINE/ONLINE mode:

- `%SW60.2 = 1`
  Controller B goes to the ONLINE mode.
- `%SW60.2 = 0`
  Controller B goes to the OFFLINE mode.

NOTE: The Primary CPU controller goes to RUN OFFLINE only if the secondary CPU is RUN Standby.

At Startup of the Secondary PLC, the secondary CPU goes to ONLINE mode (RUN Standby) only if both bits `%SW60.1` and `%SW60.2` are set to 1 (regardless of A/B assignment).

If bits `%SW60.1` and `%SW60.2` are set to 0 simultaneously, a switchover occurs:

- the Primary CPU controller goes to the RUN OFFLINE mode
- the Standby CPU goes to the RUN Primary CPU mode

To complete the switchover, bits `%SW60.1` and `%SW60.2` must be set back to 1. This makes the Offline CPU go back to the online RUN Standby mode.

The OFFLINE/ONLINE modes controlled by `%SW60.1` and `%SW60.2` are not linked to the LCD Keypad ONLINE/OFFLINE mode *(see page 227)*.

⚠ WARNING
UNEXPECTED EQUIPMENT BEHAVIOR Ensure that your system does not switchover from the application program before starting a CCOTF modification. **Failure to follow these instructions can result in death, serious injury, or equipment damage.**

System Bit %SW60.3

Application mismatch *(see page 181)*:

- `%SW60.3 = 0`
  If an Application mismatch is detected, Standby CPU is forced to OFFLINE mode.
- `%SW60.3 = 1`
  Standby CPU operates normally even if a mismatch occurs.

System Bit %SW60.4

Firmware upgrade:
- `%SW60.4 = 1`
  Allows the firmware to be upgraded on the Standby CPU, while the Primary CPU continues to control the process.
- `%SW60.4 = 0`
  Allows the firmware to be upgraded and stops the Primary CPU control of the process.

Upgrading allows:
- a Hot Standby system to operate with different versions of the OS running on the Primary CPU and Standby CPU
- upgrades without shutting down the process

To perform the firmware upgrade *(see page 195)*, the Standby CPU must be stopped. When started again, the Standby CPU operates again as the Standby CPU.

System Bit %SW60.5

Standby CPU initiates an application transfer:
- `%SW60.5 = 1` means Standby CPU requests an application program transfer from Primary CPU
- `%SW60.5 = 0` is default and no transfer occurs

NOTE: `%SW60.5` is a Monitor Bit.

`%SW60.5` monitors an action. Once the action occurs, `%SW60.5` returns to the default, which is zero (0).

NOTE: In the case of ONLINE application mismatch selected, the Hot Standby system needs 2 seconds to check the consistency of the application and the detection of an application mismatch (%SW61.4). Therefore the request for application transfer (%SW60.5) has to be done with a minimum delay of 2 seconds after any modification of the application.

⚠ WARNING

UNEXPECTED BEHAVIOR OF APPLICATION

When the ONLINE application mismatch option is selected, a request for application transfer (%SW60.5) has to be done with a minimum delay of 2 seconds after any modification of the application.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

System Bit %SW60.8

Swap Modbus on port 1:

- `%SW60.8 = 0`
  Address swap on Modbus port 1 when a switchover occurs.
- `%SW60.8 = 1`
  No address swap on Modbus port 1 when a switchover occurs.

System Bit %SW60.9

Swap Modbus on port 2:

- `%SW60.9 = 0`
  Address swap on Modbus port 2 when a switchover occurs.
- `%SW60.9 = 1`
  No address swap on Modbus port 2 when a switchover occurs.

System Bit %SW60.10

Swap Modbus on port 3:

- `%SW60.10 = 0`
  Address swap on Modbus port 3 when a switchover occurs.
- `%SW60.10 = 1`
  No address swap on Modbus port 3 when a switchover occurs.

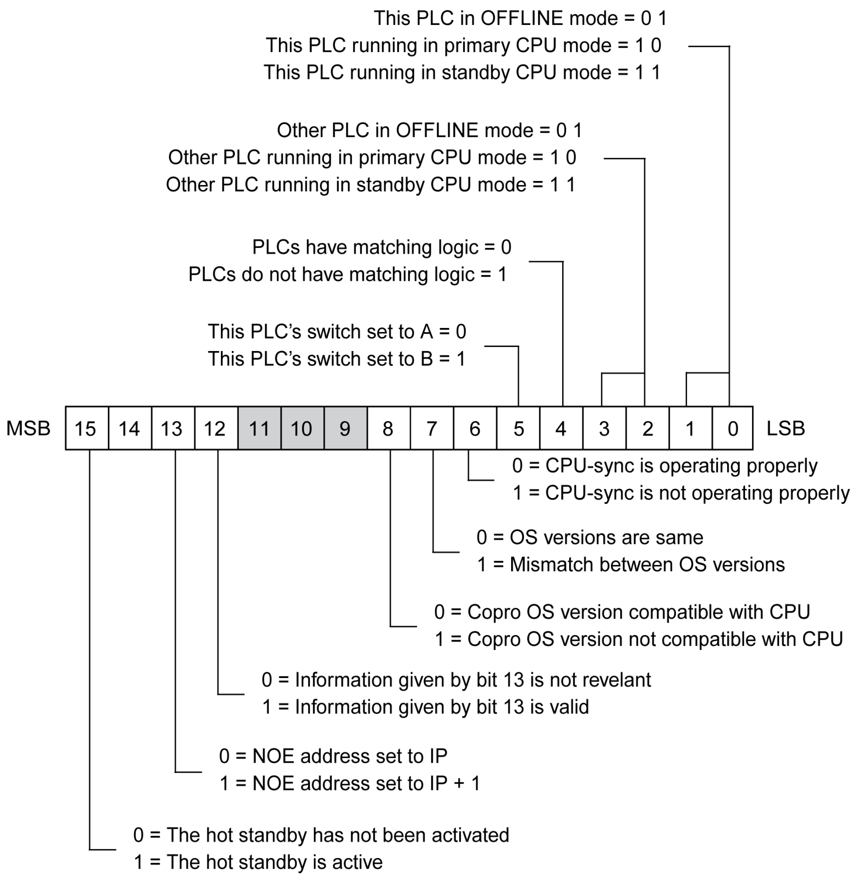
Hot Standby Status Register

Bits in the Hot Standby Status Register

The Hot Standby Status Register, System Word, `%SW61`, is read-only. It is used to monitor the current machine status of the Primary CPU and Standby CPU.

Both the Primary CPU and the Standby/Offline CPU have their own copy of the Status register. The Status register is not transferred between the Primary CPU and Standby CPU. Each PLC updates its local Status Register based on information exchanged between the two controllers.

The following illustration identifies the operating options provided by the Status Register:



System Bits %SW61.0 to %SW61.3

These four bits display the states of the local and remote Hot Standby controllers.

Status of local PLC:

- `%SW61.1 = 0` and `%SW61.0 = 1` means the local PLC is in OFFLINE mode.
- `%SW61.1 = 1` and `%SW61.0 = 0` means the local PLC is running as the Primary CPU.
- `%SW61.1 = 1` and `%SW61.0 = 1` means the local PLC is running as the Standby CPU.

Status of remote PLC:

- `%SW61.3 = 0` and `%SW61.2 = 1`means the remote PLC is in OFFLINE mode
- `%SW61.3 = 1` and `%SW61.2 = 0` means the remote PLC is running as the Primary CPU.
- `%SW61.3 = 1` and `%SW61.2 = 1` means the remote PLC is running as the Standby CPU.
- `%SW61.3 = 0` and `%SW61.2 = 0` means the remote PLC is not accessible.

System Bit %SW61.4

`%SW61.4` = 1 means that a application mismatch has been detected between the Primary CPU and Standby CPU controllers.

`%SW61.4` depends on %SW60.3 being set to 1.

System Bit %SW61.5

`%SW61.5` identifies the order reported by the Copro at start time.

The order depends on the range of the MAC addresses:

- If the A/B designation is A, then `%SW61.5` = 0.
- If the A/B designation is B, then `%SW61.5` = 1.

NOTE: The controller LCD displays either A or B.

System Bit %SW61.6

`%SW61.6` indicates if the CPU-sync link between the two PLCs is valid.

If `%SW61.6` = 0, the CPU-sync link is operating properly and the contents of `%SW61.5` is relevant.

If `%SW61.6` = 1, the CPU-sync link is not operating properly and the contents of `%SW61.5` are not relevant because the comparison of the two MAC addresses cannot be performed.

System Bit %SW61.7

`%SW61.7` indicates if the Operating Systems in the Primary and Standby PLCs are the same.

If `%SW61.7` = 0,the Operating System versions are the same in both the Primary and Standby:
- CPUs
- Copros
- Ethernet CRPs

If `%SW61.7` = 1, the two PLCs have at least one Operating System version mismatch between the Primary and Standby
- CPUs
- Copros
- Ethernet CRPs

For details about the component mismatch(s), refer to the Firmware Mismatch Register *(see page 93)*.

System Bit %SW61.8

`%SW61.8` indicates if the Operating Systems in the 2 Copros are.

If `%SW61.8` = 0, the two Copros have the same Operation System version.

If `%SW61.8` = 1, the two Copros have different Operation System versions.

System Bit %SW61.12 and 13

If `%SW61.12` = 1, the %SW61.13 indicates the address of the NOE:
- If `%SW61.13` = 1, the address is the configured IP address +1.
- If `%SW61.13` = 0, the address is the configured IP address.

If `%SW61.12` = 0, `%SW61.13` is not relevant.

System Bit %SW61.15

`%SW61.15` indicates the Copro Hot Standby activity

If `%SW61.15` = 1 means that the Copro device is set up correctly and working.

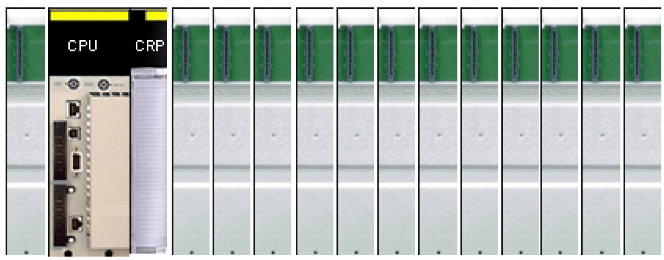
If `%SW61.15` = 0 means that the Copro device is not operating correctly.

Hot Standby Firmware Mismatch Register

Bits in the Firmware Mismatch Register (%SW69)

The Hot Standby Firmware Mismatch Register, %SW69, gives information about the firmware levels in the Primary and Standby main rack components:

- The firmware levels in the Primary and Standby CPUs, Copros and Ethernet CRPs are compared:
  - If a bit = 0, the Primary and Standby components have the same firmware versions.
  - If a bit = 1, the Primary and Standby components have different firmware versions.
- Bits 0 to 15 correspond to rack positions 16 to 1:



Slots	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
%SW69 bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		Proc mismatch	CoPro mismatch	CRP mismatch												

Using Initialized Data

Loading at Cold-Start Time

The Quantum Hot Standby CPUs support initialized data.

Initialized data allows you to specify Cold-Start initial values using Unity Pro for the data that are to be loaded at cold-start time, for more information refer to the Unity Pro Program Languages and Structure Reference Manual *(see Unity Pro, Program Languages and Structure, Reference Manual)*.

Updating Online

You can also update the initial values online, but:
- A mismatch situation occurs after updating the initial values of one CPU online in a redundant system.
- If you execute a cold-start and a switchover occurs to make the non-updated PLC the Primary PLC, the old initial values are used.

Value mismatches are treated in the same way as application mismatches. Value mismatches give the same indications and have the same update requirements as an application mismatch.

Synchronizing System Timers

Setting the System Timer in the Standby CPU Controller

In a Quantum Hot Standby system, the Primary CPU and Standby CPU controllers have their own system timers, which are not implicitly synchronized.

At Switchover, the Standby CPU sets its system timer with values sent by the Primary CPU. This mechanism allows the new Primary CPU to run the Hot Standby application in the same context as the old Primary CPU.

If the timers are not synchronized, then at Switchover, the system timer would change by the difference between the two clocks.

Non-synchronous clocks can cause problems in a time-critical application.

2.3 NOE Modules

Overview

This section describes how to use 140 NOEs, (Quantum Ethernet modules), with Unity Pro in a Quantum Hot Standby system. For a complete description of all NOE models, see the *Quantum NOE_771_xx Ethernet Modules User Guide, 840_USE_116_00*.

What's in this Section?

This section contains the following topics:

Quantum Hot Standby and 140 NOE 771 •1 Modules

Description of the Hot Standby Solution

NOTE: The Quantum Hot Standby system supports up to six 140_NOE_771_•1 Ethernet adapters.

The NOE communications modules supported are:
- 140 NOE 771 01 TCP/IP 10/100 Ethernet
- 140 NOE 771 11 TCP/IP 10/100 Ethernet

The Hot Standby NOE modules allow automatic IP address swap during a Switchover. Both controllers are configured identically. One controller is the Primary CPU NOE; the other controller is the Secondary NOE. If the Primary NOE stops, the controllers Switchover and the system recovers.

The NOE modules coordinate the swapping of IP addresses. After closing both the client and the server connections, each NOE sends a swap UDP message to its peer NOE. A sending NOE then waits a specified time-out (500 ms) for the peer swap using UDP messages. Either after receiving the messages or after a time-out, the NOE changes its IP address.

CAUTION

LOSS OF CONTROL

Use an Ethernet switch (not a hub) to connect Quantum Ethernet 140 NOE 771 •1 modules to each other and to the network to prevent a COMMUNICATION INTERRUPTION.

Failure to follow these instructions can result in injury or equipment damage.

NOTE: The NOE modules must communicate with each other to swap IP addresses. Schneider Electric recommends that you connect the Primary and Standby NOEs to the same switch because connecting two NOEs to the same switch minimizes the probability of a communication interruption.

A NOE waits for either a change in the controller's Hot Standby state or the swap of UDP messages:

If the NOE module. . .	Then. . .
Detects that the new Hot Standby state is either Primary CPU or Standby CPU	The NOE changes the IP address.
Receives a swap UDP message	The NOE transmits a swap UDP message and swaps the IP address.

All client/server services (I/O scanning, global data, messaging, FTP, SNMP, and HTTP) continue to run after the switch from the old to the new Primary CPU NOE.

NOTE: If an NOE module stops communicating, this does not cause the Primary CPU to go Offline.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

Design your application so that un-monitored modules support communication only to noncritical parts of the application.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Hot Standby and NOE Module Functionality

The 140 NOE 771 family provides different Ethernet services for the Quantum Hot Standby system.

The following table identifies the services that are available:

Service	Availably
I/O Scanning	Available
Global Data	Available
Modbus Messaging	Available
FTP/TFTP	Available
SNMP	Available
HTTP Server	Available
DHCP	Unavailable

NOTE: The 140 NOE 771 •1 modules support a Quantum Hot Standby system starting with Unity Pro V2.0.

NOE Operating Modes in Quantum Hot Standby System

The NOE Modes

The 140 NOE 771 module modes are:

- Primary NOE Mode:
  The Hot Standby mode is Primary CPU and all client/server services are active.
- Standby NOE Mode:
  The Hot Standby mode is Standby CPU and all server services are active except DHCP.
- Standalone Mode:
  The NOE is in a non-redundant system, or the CPU is not present or not healthy.
- Offline Mode:
  The CPU is stopped.

The Modicon Quantum Hot Standby and the NOE operating modes are synchronized by these conditions:

CPU Module Status	Hot Standby State	NOE Operating Mode
Present and Healthy	Primary CPU	Primary
Present and Healthy	Standby CPU	Standby
Present and Healthy	Offline	Offline
Present and Healthy	Unassigned	Standalone
Not present or unhealthy	N/A	Standalone

Any of the following events affect the NOE operating mode:

- the NOE is powered up
- an NOE executes a Hot Standby Switchover
- an NOE goes to Offline mode
- a new application is downloaded to the NOE

IP Address Assignment at Power-Up

An NOE obtains its IP address assignment at power-up as follows:

If the Hot Standby state is...	Then the IP address assigned is...
unassigned	IP address configured in Unity Pro
Primary CPU	IP address configured in Unity Pro
Standby CPU	IP address configured in Unity Pro + 1
unassigned to offline transition	See *Offline Mode at Power-up Sequence* in the next table.

If two NOEs power-up simultaneously, a resolution algorithm:
- determines the Primary NOE
- assigns the IP address configured in Unity Pro to that Primary NOE
- assigns the IP address configured in Unity Pro + 1 to the Standby NOE

Offline Mode at Power-up Sequence	**Result**
Controller A powers-up before controller B	• IP address of controller A is the IP address configured in Unity Pro • IP address of controller B is the IP address configured in Unity Pro + 1
Both controller A and controller B power-up at the same time	The resolution algorithm assigns the configured IP address to controller A, and it assigns the configured IP address + 1 to controller B.

The NOE performs a duplicate IP test by issuing an ARP request to the IP address configured in Unity Pro. If a response is received within 3 seconds, the IP address remains at the default IP and the NOE blinks a diagnostic code.

If no IP configuration exists, the NOE remains in the Standalone mode and the IP address must be obtained from either a BOOTP server or the MAC address.

Ethernet Services at Power-Up

The following table shows how the status of an NOE service is affected by the Quantum Hot Standby state:

Hot Standby State	**Status of NOE Services**					
	Client Services		**Client/Server Services**	**Server Services**		
	I/O Scanner	**Global Data**	**Modbus Messaging**	**FTP**	**SNMP**	**HTTP**
Unassigned	Run	Run	Run	Run	Run	Run
Primary CPU	Run	Run	Run	Run	Run	Run
Standby CPU	Stop	Stop	Run	Run	Run	Run
Offline	Stop	Stop	Run	Run	Run	Run

Hot Standby Switchover

The following table describes how the NOEs coordinate a Hot Standby Switchover:

Step	Action
1	In a Hot Standby configuration NOE A is running in the Primary PLC and NOE B is in the Standby PLC.
2	NOE A detects that its PLC has changed from Primary CPU to the Offline mode.
3	NOE A changes from Primary NOE to Offline with the same Ethernet services running and starts its watchdog timer (with a 500 ms time-out setting). It waits for a UDP request to swap IP addresses from NOE B.
4	NOE B detects that its PLC has changed state from Standby PLC to Primary CPU.
5	NOE B stops all its Ethernet services, sends a UDP request to NOE A for the synchronization of the IP address swap, starts its watchdog timer (with a 500 ms time-out setting) and waits for an UDP response from NOE A.
6	When NOE A receives the UDP request from NOE B (or after the NOE A watchdog timer times out), it stops all its Ethernet services: ● If it has received a UDP request, NOE B sends a UDP response to NOE A. ● If its watchdog timer has timed out, NOE B does not send a UDP response. NOE A then swaps its IP address and starts the Secondary services.
7	NOE B swaps IP addresses and starts Ethernet services as Primary NOE.
8	After NOE A senses that its local CPU changes from Offline to Standby, it takes the Secondary IP address.
9	NOE B now becomes the Primary NOE.
10	NOE B opens all client connections, listens for all server connections and reestablishes those connections.
11	NOE A listens for all server connections and reestablishes those connections.

NOTE: During the Hot Standby switchover, there is a loss of communication during 500 ms between the PLC and the HMI and/or Unity Pro.

Going to Offline

When either the CPU stops or the Hot Standby CPU goes to Offline mode, two events occur:

1. NOE goes to the Offline mode
2. NOE uses the IP address of the present configuration

The IP address assignment when going offline:

Hot Standby State	**IP Address Assigned Is...**
Primary CPU to Offline	Configured IP address, if other controller **does not** go to Primary CPU mode
Standby CPU to Offline	Configured IP address + 1

NOTE: For more information, refer to NOE IP Address Assignment *(see page 103)*.

NOE IP Address Assignment

Configuring a 140 NOE 771 •1 Module

Since the Primary and Standby PLCs in Quantum Hot Standby system must have an identical configurations, the configured NOE IP addresses are the same. The current local Hot Standby mode determines the IP address.

This table shows how the NOE IP addresses are assigned:

Hot Standby State	IP Address
Primary CPU	IP address configured in Unity Pro
Standby CPU	IP address configured in Unity Pro + 1
Transition from Primary to Offline	IP address configured in Unity Pro, if peer controller **does not** go to Primary
Transition from Standby to Offline	IP address configured in Unity Pro + 1

IP Address Restrictions

Do not use either the **broadcast IP address** or **broadcast IP address - 2** to configure a NOE module.

Do not configure the Primary CPU address as nnn.nnn.nnn.254. This causes the Standby CPU IP address to be: nnn.nnn.nnn.255. The Standby CPU would then return the diagnostic code **Bad IP configuration**.

IP Address Transparency

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

For a Quantum Hot Standby configuration:
- Do not use the IP address configured in Unity Pro + 1.
- Do not use consecutive IP addresses of the IP address configured in Unity Pro.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

When a Switchover occurs, the new Primary PLC takes the IP address of the old Primary PLC.

When the PLC that has stopped becomes operational again and rejoins the Hot Standby system, it takes the IP address of the Standby PLC.

The new Primary NOE must have the same IP address as the former Primary NOE. The IP address in the Secondary NOE is IP address + 1.

The NOEs integrated into the Quantum Hot Standby configuration coordinate this swapping of IP addresses with the management of Ethernet services used.

NOE IP Address Swap Time

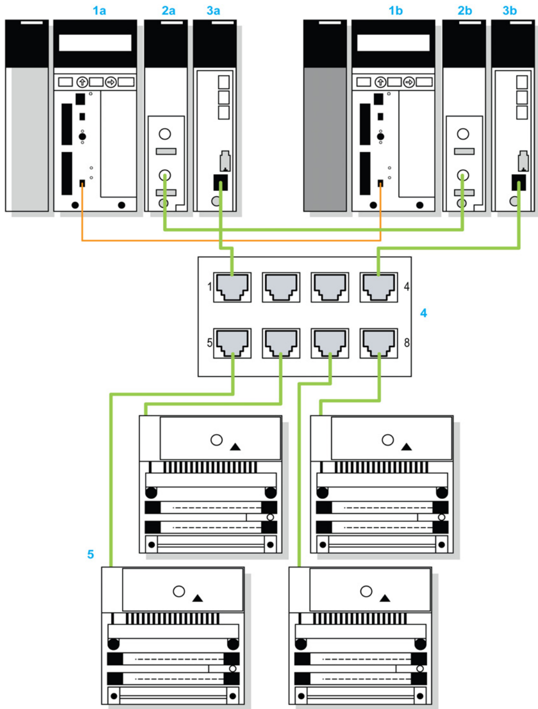
The following table details the 140 NOE 771 •1module address swap time:

Typical Swap Time	Maximum Swap Time
6 ms	500 ms

NOE Modules in Hot Standby System

DIO Ethernet Topology

A Quantum Hot Standby system can control distributed I/O devices using NOE modules. The example below contains no Ethernet RIO modules:



Legend:
1a Primary CPU
2a Primary CRP module
3a Primary NOE module
1b Standby CPU
2b Standby CRP module
3b Standby NOE module
4 Ethernet switch
5 Distributed I/O devices

SNMP

An SNMP service on the NOE module is always active, that is not linked to the Hot Standby mode.

Overloaded Network

Overview

If an NOE 771 xx is used to run in a 100Mb/s Ethernet and a persistent overloaded network occurs, the NOE 771 may go into Kernel mode. This could cause the Primary CPU to go to STOP Offline.

Example

An example of a persistent overloaded network would be when two ports of an Ethernet switch are linked to each other: this would be seen by all Ethernet nodes connected to the sub-network and result in a massive overloaded network (that does not occur on properly configured network).

NOTE: Broadcasts and especially ARPs, are part of standard Ethernet traffic and will have no adverse effects on an NOE. Even "small" storms that take up to 5% of the basic network traffic over short periods (from several seconds to 2-3 minutes) would not overload the NOE. It is only the massive and enduring overloaded network (such as those created by a looped network cable) that can cause problems for the Hot Standby system with NOEs.

Impact on CPU

For backplane communication, the NOE has direct access (DMA) to the memory of the CPU module. Therefore, if the NOE goes into Kernel mode while accessing the CPU, this may have an impact on the CPU behavior. In rare cases, it can even cause the Primary CPU to go to STOP Offline. In this case, the Standby CPU will take the hand as the Primary CPU.

Recommended Actions

Take the following steps to protect against the unwanted effects of excessive broadcast traffic:

- Reduce the speed of the port allocated to communicate with the respective NOEs from 100Mb/s to 10Mb/s.
- Limit the potential effects of an overloaded network to the NOE by filtering it with an appropriate switch set, with a limit of 500 packets per second.
  (Schneider Electric offers a line of a configurable ConneXium switches, capable of broadcast limiting.)
- If the Ethernet switch must be set at 100Mb/s speed, then set the watchdog timer to 1.5 seconds (independent of the number of NOEs). If the watchdog timer is set too low, then the remaining system may also stop working if a persistent overloaded network occurs.

Maintaining a Quantum Hot Standby System

3

Overview

This chapter provides information about maintaining a Quantum Hot Standby system with Unity Pro.

What's in this Chapter?

This chapter contains the following sections:

3.1 Hot Standby Module Replacement

Replacing a Module

Replacing a Module without Stopping

A module to be replaced must be in the Standby PLC in the Offline mode. If the inoperative module is in the Primary PLC, perform a Switchover to move the module to the Standby mode, then move the Standby PLC in the Offline mode.

The Offline mode ensures that the system does not try to do a Switchover while replacing a module. The Primary PLC continues to control the system as a non-redundant Standalone PLC during the module replacement.

Ensure that the new module being replaced:
- resides in the same position as in the Primary backplane
- is the identical type of module as the module to be replaced

⚠ WARNING

UNEXPECTED EQUIPMENT BEHAVIOUR

Do not remove a module from the Primary PLC that is under power (Hot Swapping is not allowed in the Primary PLC of a Quantum Hot Standby system).

Failure to follow these instructions can result in death, serious injury, or equipment damage.

3.2 Hot Standby Health Messages

Verifying the Health of a Quantum Hot Standby System

Redundant Links

Because there are two links (Copro and RIO) between the Primary and Standby PLCs, if a PLC detects an error on one link, it still has a link available to send diagnostic information to the other PLC.

Generating and Sending Health Messages

The Quantum Hot Standby modules exchange a health message approximately every 10 ms.

If the Primary CPU becomes inoperative, the Standby CPU is notified and assumes the Primary CPU role.

However, if the Standby CPU becomes inoperative, the Primary CPU continues to operate as a standalone CPU.

The RIO head processors periodically verify communication with one another.

The Primary CPU sends a health message to the Standby CPU either:
- every 10 milliseconds when no other data is being sent on the Copro link
- every 5 milliseconds if no communication is required with any drop on the RIO link.

If the Standby CPU never receives any message on either of these links (Copro, S908 RIO or Quantum Ethernet I/O Ethernet RIO), the Standby CPU tries to determine the cause and assumes control if necessary.

If the Primary CPU does not receive a valid response from the Standby CPU, the Primary CPU operates as a standalone CPU.

Conducting Startup Tests

The system automatically performs startup confidence testing on the Quantum Hot Standby Copro that attempt to detect hardware errors in the Copro before the application is allowed to run.

If the Copro does not pass the tests, the Standby PLC remains offline and does not communicate with the other Quantum Hot Standby modules.

Conducting Run Time Tests

The system automatically performs run time tests whenever the Copro is in the operational state.

NOTE: Run time tests are executed in small slices to prevent delays in scan time.

If the Copro does not pass the tests, the Standby PLC remains offline and does not communicate with the other Quantum Hot Standby modules.

3.3 Single Point of Detected Failure

Overview

This section describes the location of various detected errors that can cause a Quantum Hot Standby to change to a standalone system or become inoperative.

What's in this Section?

This section contains the following topics:

Detecting and Diagnosing Inoperative Components through Health Messages

Inoperative Components

An inoperative component causes system changes:

If...	Then...
A component of the Primary CPU becomes inoperative	Control shifts to Standby CPU
A component of the Standby CPU becomes inoperative	Standby CPU goes offline
Fiber optic Sync-link cable link becomes inoperative	Standby CPU goes offline

Health Messages

The Primary CPU sends a health message to the Standby CPU over Copro Sync-link every 10 milliseconds.

Health messages over the RIO link:

If...	Then Primary CPU Sends Health Message over RIO link...
No communication is required with any drop on RIO link	Every 5 ms
All systems are OK	Every scan

Lack of health messages over the RIO link:

If...	Then...
Standby CPU does not receive health messages on neither the Copro link nor the RIO link	**1.** Standby CPU determines the cause **2.** Standby CPU assumes control by becoming the Primary CPU
Primary CPU does not receive a valid response from the Standby CPU	Primary CPU operates as a non-redundant Standalone PLC.

Finding Diagnostic Information with Unity Pro

Detected errors and Switchovers are logged in the diagnostic buffer. To view the buffer log select from the menu **Tools →Diagnostic Viewer**.

Detected Inoperative Conditions on Rack, CPU, Copro and RIO Head

Communication Timeouts

On every scan, the transfer of data between Primary and Standby CPUs insures that they are synchronized. Timers in this communication are the first level of error detection:

1. The Primary CPU waits for the Standby acknowledgement. A timeout here is due to an inoperative:
   - Primary Copro
   - Standby CPU
2. The Standby CPU waits for the Primary acknowledgement. A timeout here is due to an inoperative:
   - Standby Copro
   - Primary CPU
3. The Primary Copro waits for the Standby acknowledgement. A timeout here is due to an inoperative Standby PLC.

CPU Sync-Link Interruption

There are 3 possible cases:

- Copro-Copro link interruption
  This condition is detected by both Copros. The Standby PLC detects the interruption and goes Offline. The Primary PLC detects that the Standby PLC has disappeared, reports it to the log and continues to scan the I/O as a Standalone PLC
- Primary Copro inoperative
  This condition is not detected, the Primary CPU continues to scan the I/O, but as a Standalone PLC. The Standby PLC goes Offline.
- Standby Copro inoperative
  This condition is detected by both Copros. The Standby PLC goes Offline. The Primary PLC detects that the Standby PLC has disappeared, reports it to the log and continues to scan the I/O as a Standalone PLC.

NOTE: The Primary CPU maintains continuous activity on link, which allows the Standby CPU to detect a comminutions interruption as soon as possible.

Inoperative Rack

There are 2 possible cases:

- Inoperative Primary rack
  The Standby PLC detects that the Primary PLC has disappeared and takes control of the system. It scans the I/O as a Standalone PLC.
- Inoperative Standby rack
  The Primary PLC detects that the Standby PLC has disappeared, reports it to the log and continues to scan the I/O as a Standalone PLC.

Copro Inoperative

The high speed CPU Sync-link connects the Primary and Standby Copros. The primary CPU communicates with the Standby CPU every 10 ms with either a:

- data message
- health message

The Primary Copro waits for an acknowledgement from the Standby Copro.

Detecting Copro errors:

If ...	Then ...
Primary Copro reports a detected error to the Primary CPU	Primary CPU controller: **1.** acknowledges the detected error **2.** attempts to transfer control to the other controller by sending a `take control` command to the Standby CPU through the RIO link
Primary Copro does not respond within 5 ms to the Primary CPU	Primary CPU controller: **1.** detects and acknowledges the error **2.** attempts to transfer control to the other controller by sending a `take control` command to the Standby CPU through the RIO link
Primary CPU Copro sends a `take control` command to the Standby Copro	Primary CPU Copro: **1.** relinquishes control and becomes the Standby CPU **2.** does not expect any response
Standby Copro reports a detected error to the Standby CPU	Standby CPU controller: **1.** reports the error by sending a No Standby CPU message **2.** goes offline

Inoperative S908 CRP RIO Head

There are 2 cases for inoperative S908 CRPs:

- inoperative Primary CRP
  This condition is detected by both the Primary and Standby PLCs. The Standby PLC takes control of the system. The Primary Copro goes offline.
- Inoperative Standby CRP
  This condition is detected by the Standby PLC, which reports the condition to the Primary PLC and then goes offline.

Inoperative Quantum Ethernet I/O Ethernet CRP RIO Head

There are 2 cases for inoperative Ethernet CRPs:

- Inoperative Primary CRP
  This condition is detected by both the Primary and Standby PLCs. The Standby PLC takes control of the system and scans the I/O, but as a Standalone PLC. The Primary goes Offline.
- Inoperative Standby CRP
  This condition is detected by the Standby PLC and Primary Copro, which reports the condition to the Primary PLC.The Standby PLC goes Offline. The Primary continues to scan the I/O, but as a Standalone PLC.

RIO Link Operations

The Primary CPU sends a health message about its RIO CRP Head link to the Standby RIO Head every 5 ms.

Inoperative S908 RIO Link

There are 3 cases of an inoperative S908 RIO link:

- interrupted link from Primary CRP Head
  This condition is detected by the Standby CRP Head. The Primary Copro goes Offline. The Standby PLC takes control of the system and scans the I/O as a Standalone PLC.
- interrupted link from Standby CRP Head
  This condition is detected by the Standby CRP Head and the Standby PLC goes Offline. The Primary PLC continues to scan the I/O, but as a Standalone PLC.
- interrupted RIO CRA Drop
  This condition is not detected by the Quantum Hot Standby system.

Inoperative Quantum Ethernet I/O Ethernet RIO Link

This condition is detected by the both Primary and Standby CRPs.

If the Standby CRP detects an inoperative Quantum Ethernet I/O RIO network (it cannot communicate with the Primary CPU), the Standby CPU requests the Primary CPU to check RIO network via its Copro:.

- if the Primary CPU is operational, it checks the RIO connection:
  - if the connection is OK, the Primary CPU continues to control the system and the Standby CPU goes to RUN Offline
  - if the connection is inoperative, there is a Switchover. The Standby CPU takes control of the system and the Primary CPU goes to RUN Offline
- if the Primary CPU is inoperative, the Standby CPU takes control of the system

There are 2 cases:

- If the user application **does not have** the Link Redundancy Needed FB implemented:
  An inoperative RIO network is detected by both Primary and Standby Quantum Ethernet I/O Ethernet CRPs. the Standby PLC goes Offline while the network repairs itself. When the network works again, this PLC goes back online as the Standby PLC again.
- If the user application **does have** the Link Redundancy Needed FB implemented:

Detecting High Speed Sync-Link Interruptions

Diagnostic Information

Facts

1	High-speed data link connects the two Copros.
2	Using the high-speed data link, the Primary CPU controller communicates with the Standby CPU every 10 milliseconds.
3	Primary CPU sends either • data message • health message

NOTE: If both the Primary CPU and Standby CPU do not hear from each other, either station can detect a high speed data link interruption.

Standby CPU Detects an Error

At first,

Step	Action	Result
1	Standby CPU does not hear from the Primary CPU on the high-speed data link	**1.** Standby CPU requests the Primary CPU to monitor the RIO link **2.** Primary CPU sends a request to the RIO Head

When the RIO Head receives the request,

If ...	Then ...
RIO Head finds the RIO link not active	**1.** RIO Head assumes that the Primary CPU must be down **2.** Standby CPU assumes control
RIO Head finds the RIO link is active	Message received from Primary CPU must be either **1.** health message Messages are sent every 5 milliseconds from Primary CPU RIO Head to Standby CPU RIO Head. **2.** I/O transaction data message Messages are sent from the Primary CPU RIO Head to the I/O drops at the request of the controller.

Facts about the I/O

1	If the message is an I/O transaction, the RIO Head **1.** concludes an interruption occurred on the high-speed data link **2.** informs the Primary CPU controller to go to offline
2	If you never configure an I/O drop, the high-speed data link could cause the Standby CPU to assume control since the Standby CPU RIO head will never receive any I/O transaction message.
3	After any CPU error is detected , **1.** RIO Head will not perform drop communication **2.** RIO Head sends only health messages

Standby CPU Assumes Control

The Standby CPU becomes Primary CPU

Step	Action	Result
1	After the Primary CPU controller goes offline,	A health message from the Standby CPU controller is the only message received by the Standby CPU RIO Head.
2	Standby CPU controller listens to the high-speed data link for one scan.	
3	If Standby CPU controller hears nothing,	Standby CPU knows that the cause must be on both the Primary CPU Copro and Primary CPU.
4	Standby CPU assumes control.	

Troubleshooting Primary PLC

Overview

To determine which component has become inoperative note the:
- controller status displayed in the CPU LCD screen
- RIO Head status displayed in the RIO Head LED screen

Troubleshooting the Primary CPU

This table gives the location of Primary PLC detected errors:

Controller Status	RIO Head Status	Detected Error Type	Description
Stop	All LEDs off except **Ready** on and **Com Act** blinks four times	Controller	A detected interface error occurred.
Offline	All LEDs off except **Ready** on	Fiber Optic connection between PLCs	A communication error was detected.
Stop	All LEDs off except **Ready** on and **Com Act** displays a detected error pattern *(see page 215)*	RIO Head	A communication error was detected.
Stop	**Ready** on and **Com Act** blinks four times	RIO Cable becomes inoperative at Primary CPU End	In a dual cable system, if only one cable is inoperative, the **Error A** or **Error B** LED on the RIO Head lights up instead of stopping the system. **NOTE:** When the RIO cable becomes inoperative at the Primary CPU end, the input data may be reset to 0 for one scan because the communication interruption to the drop occurs before the broken link is detected.

NOTE: In a Quantum Hot Standby configuration without RIO drop, the A and B detected error LEDs are not relevant when using CRP module with a firmware version lower than 2.00.

Troubleshooting the Standby CPU

This table gives the location of Standby PLC detected errors:.

Controller Status	RIO Head Status	Detected Error Type	Description
Stop	All LEDS off except **Ready** on or **Ready** on and **Com Act** blinks once a second	Controller	A detected Interface error occurred.
Offline	**Ready** on and **Com Act** stops blinking	Fiber Optic connection between both controllers	A detected communication error occurred.
Stop	**Com Act** displays detected error pattern *(see page 215)*	RIO Head	After replacing the module and cycled power, to ensure that the controllers have identical application programs, perform an application program update.
Stop	**Ready** on and **Com Act** blinks four times	RIO Cable becomes inoperative at Standby CPU end	In a dual cable system, the RIO Head gives no indication if only one cable has become inoperative.
Offline	**Com Act** on	Either type of fiber link interruption: • from Standby CPU Transmit to Primary CPU Receive • from Primary CPU Transmit to Standby CPU Receive	

Programming and Debugging

4

Overview

This chapter describes what is necessary to know to program and debug applications for a Quantum Hot Standby system.

What's in this Chapter?

This chapter contains the following sections:

4.1 Operating Modes and Switchover Information

Overview

This section describes the Quantum Hot Standby operating modes, switchover behavior and performance.

What's in this Section?

This section contains the following topics:

Operating States and Modes

Description of the Hot Standby States

- **Run Primary CPU**
  The Primary CPU PLC executes the application program and updates the remote I/Os. If a Standby CPU is present, the Primary CPU sends application data and I/O to it.
- **Run Standby CPU**
  During each cycle the PLC:
  - checks that a Primary PLC exists
  - checks that there is no command from the Primary PLC
  - indicates to the Primary CPU that it is running well and is ready to take over the process if the Primary CPU stops
  - checks that there is no CPU, Copro or CRP mismatches (unless allowed)

  Local I/Os are updated, but not the Remote I/Os.
  For a Quantum Ethernet I/O configuration, the Standby PLC checks the RIO Drop connected.
- **Run Offline**
  Depending on the setting of Behavior of the CPU in Run Offline mode *(see page 78)* the PLC executes:
  - all sections of the MAST task application program but the I/O is not written
  - the first section of the MAST task of the application program but the I/O is not written
  - none of the application program MAST task

  This state is either manually activated or by the CPU, which detects the state by itself.
  If there is no Primary PLC, the CPU tries to change to the Run Primary CPU state.
  If the Primary PLC exists, the PLC checks each cycle to see if can go to the Run Standby CPU state. There are several commands are available:
  - Application transfer
  - Any online command
  - STOP command
  - HALT command
- **Stop (Offline)**
  The PLC neither executes the application program nor controls the process. It is not part of the Hot Standby system. Two commands are available
  - Application transfer
  - RUN command
  - Init

The run offline and the stop offline state can occur in the Primary CPU and in the Standby CPU at the same time.

Table of States

The following table shows the possible states of the 2 controllers of a Hot Standby configuration:

		Controller A state			
		Run Primary	**Run Standby**	**Run OffLine**	**Stop OffLine**
Controller B state	Run Prim	N/A	Hot Standby active I/O processed	Hot Standby inactive I/O processed	Hot Standby inactive I/O processed
	Run Standby	Hot Standby active I/O processed	N/A	N/A	N/A
	Run OffLine	Hot Standby inactive I/O processed	N/A	Hot Standby inactive I/O not processed	Hot Standby inactive I/O not processed
	Stop OffLine	Hot Standby inactive I/O processed	N/A	Hot Standby inactive I/O not processed	Hot Standby inactive I/O not processed

Description of Run Offline Use Cases

The following table describes the different situations of the Run Offline state:

If ...	Then ...
The Primary CPU PLC enters Run Offline state	The Standby CPU PLC takes over the process and becomes Run Primary CPU
The Standby CPU PLC enters Run Offline state	The Hot Standby function is no longer available
The fibre optic link is disconnected	The Standby CPU PLC enters Run Offline state
The actual hardware configuration is different from the configuration defined in the project	Either the Primary CPU or the Standby CPU PLC starts in Run Offline state
A application mismatch occurs	The Standby CPU PLC enters Run Offline state
The Standby CPU RIO head (CRP) stops operating	The Standby CPU PLC enters Run Offline state
There is no RIO connection open	Either the Primary CPU or the Standby CPU starts in RUN OFFLINE mode

RUN OFFLINE State Recommendation

In the RUN OFFLINE state the PLC is not configured as a PRIMARY nor a STANDBY CPU. This occurs after the Hot Standby system detects an error or Hot Standby OFFLINE mode has been chosen.

In this state, the CPU main actions are:

- execution of the code sections depending on the choice in the CPU Executes *(see page 78)* menu
- no data transfer from primary, except for the %SW60 value

- address swap management
- local IO management

When using communication EFBs, some applications can be affected by the entire code execution.

It is recommended to:
- create a boolean variable
  `cpu_state:=(%SW61.1) AND NOT (%SW61.0);`
- assign the section or communication block execution to this variable

With such a fix, unexpected EFB communication calls are avoided if the Standby CPU goes to an OFFLINE state.

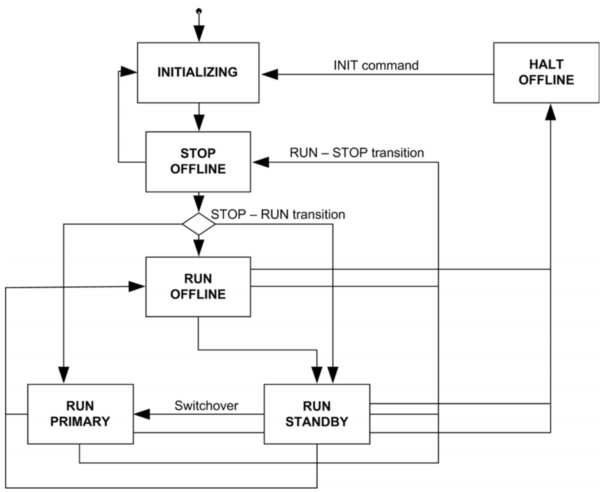
Recovery from RUN OFFLINE when ERIO is used

To recover from RUN OFFLINE due to no-Drop condition:
- cable the drops
- put one CPU of the system in the STOP mode
- wait until at least one Drop has opened a connection with CRP Head

To put the system in the RUN mode again, carry out the procedure in Quantum Hot Standby Operating Modes Overview *(see page 44)*.

Description of the Hot Standby Operating Modes

A Quantum Hot Standby PLC has some restrictions in terms of changing modes. The following figure shows the state diagram of the Hot Standby Quantum system:



NOTE:

- A PLC that is in the Run Offline mode cannot go directly to Run Primary Mode
- A PLC that is in the Run Primary mode cannot go directly to Run Standby Mode

Automatic Start in Run Option

At a cold start with the **Automatic Start in Run** option configured, a PLC restarts depending on the operating mode of the other PLC, this PLC's operability and on whether identical applications are present on both PLCs:

If...	Then...
The other PLC is Primary, the two applications are identical and this PLC is operating normally	The PLC restarts in Standby mode
The other PLC is Primary and the two applications are not identical or this PLC is not operating normally	The PLC restarts in Offline mode
There is no Primary and this PLC is operating normally	The PLC restarts in Primary mode
There is no Primary but this PLC is not operating normally	The PLC restarts in Offline mode

A local abnormal or inoperative operating state will be reported when:
- There is a loss of power to the CPU rack
- An application program error that generates a HALT state (for example, a blocking software error)
- The hardware or firmware of the CPU module becomes inoperative
- The CPU-sync link is disconnected

At a warm start, the PLC restarts depending on the previous PLC operating mode (Stop or Run).

If the previous state was Run, the PLC restarts according to:
- operating mode of the other PLC
- local PLC's operability or non-operability
- on whether identical applications are present on both PLCs (refer to the table above)

System Performances

Switchover Time

A Switchover between the time of the detection of the event that causes a Switchover until the Standby controller takes control is less then one CPU cycle.

This cycle time is defined:

- for “cyclic” cycle times, by the MAST watchdog period of time
- for “periodic” cycle times, by the MAST period of time

Application Response Time

Normally the Hot Standby system application response time (ART) is the same as in a Standalone system.

But the ART is increased by:

- 1 MAST watchdog if the Switchover is due to an event in the Primary CPU. This increase is due to execution in the “new” Primary CPU of the instructions that was being executed in the “old” Primary CPU before the Switchover.
- up to 1 MAST cycle if the Switchover is due to a user command. The amount of ART increase depends on the amount of data transferred.

Conditions for Switchover

Commanding Manual Switchovers

In addition to the system conditions *(see page 113)* that cause an automatic Switchover, a manual Switchover can be commanded by:

- writing to bits 1 and 2 of the Unity Command Register at %SW60 *(see page 90)*. This write operation can be accomplished by:
  - the application
  - issuing a Modbus request from a remote HMI
  - Unity Pro animation table
- sending a RUN -> STOP command from Unity Pro to the Primary CPU
- Offline command from the Primary CPU keypad

NOTE: Before doing any switchover by application program, ensure that the Standby PLC is ready to assume the Primary role. Refer to the Unity Pro Program Languages and Structure Reference Manual *(see Unity Pro, Program Languages and Structure, Reference Manual)* for more information about the %SW182-%SW183 and %SW176-%SW177 system words.

NOTE: The intended use of user application Switchover (in %SW60) is to react to. detected error by the application. Do not use this method for periodic Switchovers.

NOTE: If for some reason the application has to Switchover periodically, the period between switchovers must not be less 120sconds.

Example of Switchover with PLC B Initially in Standby Mode

In this example, the initial state of the system is as follows:

- PLC A has a RUN command (%SW60.1 = 1) and is acting as the Primary
- PLC B has a RUN command (%SW60.2 = 1) and is acting as the Standby

By writing new values to bits 1 and 2 of the %SW60 command register, you can command a change in the operating modes of the Hot Standby controllers. The following table describes the four commands and their results:

New Values Written to %SW60		**Resulting PLC Operating Modes**		**Effects**
Bit 1	**Bit 2**	**PLC A**	**PLC B**	
0	0	Offline	Standby ↓ Primary	• **Switchover event is immediate** • System remains redundant
0	1	Offline	Standby ↓ Primary	• **Switchover event occurs** within one MAST task* • System is no longer redundant

New Values Written to %SW60		Resulting PLC Operating Modes		Effects
Bit 1	**Bit 2**	**PLC A**	**PLC B**	
1	0	Primary	Offline	• **No Switchover event** • System is no longer redundant
1	1	Primary	Standby	• **No Switchover event** • No change from initial conditions
* In this case, we are not directly commanding a Switchover. Instead, we are commanding PLC A to enter an Offline state and we are relying on the system logic to recognize this and change PLC B from Standby to Primary during the next scan.				

NOTE: All changes to the command register %SW60 must be written to the Primary PLC. This register is copied from the Primary to the Standby PLC during each MAST task. Therefore, any changes you make directly to the Standby PLC's command register will be overwritten by this transfer without taking effect.

Switchover Behavior during Application Mismatch

Modifying the Application Variables

If a switchover occurs during application mismatch, the new Primary CPU executes its own, different application program with the data received from the other controller.

Depending on the modification, different behaviors occur:

Modification	Effect
Only code changed (same variables)	All the variables exchanged between the controllers are equal.
Variables added to the initial Primary CPU	Variables are not used by the new Primary CPU.
Variables deleted from the initial Primary CPU	New Primary CPU executes application program using the latest values for these variables.
Variables added to the initial Standby CPU	New Primary CPU executes application program using initial values for these variables.
Variables deleted from the initial Standby CPU	New Primary CPU does not use these variables

Modifying an SFC Section with Unity Pro

Schneider Electric recommends not using the SFC programming language in a Hot Standby application.

NOTE: The SFC programming language is not available for 140 CPU 671 60S Hot Standby applications.

NOTE: Modification of existing SFC actions and transitions have no impact on the SFC execution. A Switchover does not reset the SFC to its initial step.

If SFC is used in a Hot Standby application, system reaction to online modifications depends on the setting of %SW60.3 *(see page 87)*, the logic mismatch bit:

- If a mismatch **is not** allowed, SFC modifications do not cause a problem. When the Primary CPU application changes, the Standby CPU goes to the RUN OFFLINE mode. the Primary CPU application must be transferred to the secondary CPU to move it back to the RUN STANDBY mode.

NOTE: A transfer carried out automatically by the application reduces the time that the Hot Standby function is not available to o a minimum.

- If a mismatch **is** allowed, then:
  - An SFC modification may cause a reallocation of the block containing the SFC data. This stops the exchange of this data with the Standby CPU.
  - Also, after a Switchover, this SFC restarts at its initial step. This may have an impact on the operation of the Hot Standby application.
  - To reduce these impacts, program the SFC in several sections. The modification of one SFC does not the rest of the SFC.

⚠ WARNING
UNINTENDED EQUIPMENT OPERATION
Ensure that the controllers contain the same application program during a Switchover.
Failure to follow these instructions can result in death, serious injury, or equipment damage.

If a Switchover occurs in the Run mode and there is an application mismatch between the two controllers, the Standby CPU assumes Primary CPU responsibilities and starts executing a different application program from the previous Primary CPU.

Remove any application mismatch by performing an Application Program Transfer *(see page 160)* as soon as possible after completing modifications.

Handling Network Addresses at Switchover

Overview

The following material describes handling network addresses at Switchover. A Quantum Hot Standby system can communicate data over different network protocols:
- Modbus
- Modbus Plus
- TCP/IP

⚠ WARNING

UNEXPECTED EQUIPMENT OPERATION

Offset address must not be assigned to another device than the peered PLC of the Hot standby system.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

In Hot Standby applications, correct address assignment must be fulfilled for proper operation of network address swap at switchover.

Swapping Modbus Addresses at Switchover

In a Quantum Hot Standby system, the Modbus port addresses are:
- Primary CPU: 1-119
- Standby CPU: Offset +128
- Maximum address: 247
- Range 1-247

The Modbus port addresses can be changed using one of two methods:
- Communication menu in the front panel keypad
- Modbus Port tab in the Unity Pro editor

Changing addresses:

<table>
<tr><th colspan="2">Using the Communication menu in the Front Panel Keypad</th></tr>
<tr><td colspan="2">Change address on either:</td></tr>
<tr><td>Primary CPU:
1. Access the front panel keypad of the Primary CPU.
2. Go to Communication menu.
3. Go to Serial Port submenu.
4. Select address.
5. Change address.
6. Perform application program transfer.
7. Verify Standby CPU Modbus address is +128.</td><td>Standby CPU:
1. Access the front panel keypad of the Standby CPU.
2. Go to Communication menu.
3. Go to Serial Port submenu.
4. Select address.
5. Change address.
6. Perform Switchover.
7. Ensure Standby CPU switched to Primary CPU.
8. Perform application program transfer.
9. Verify Standby CPU Modbus address is +128.</td></tr>
<tr><th colspan="2">Using the Modbus Port Tab in Unity Pro Editor</th></tr>
<tr><td>To change address, download application program. (see page 189)</td><td></td></tr>
<tr><td colspan="2">Note: If the Modbus address is changed in the Primary CPU using the front panel keypad, ensure that application program transfer is made to enable the corresponding Modbus Switchover in the Standby CPU.</td></tr>
</table>

NOTE: In a Quantum Hot Standby system only one port is available for Modbus.

By default, there is an address swap at Switchover between the Primary CPU and Standby CPU Modbus ports. This default condition can be changed using the following two methods:

- Using Hot Standby menu in the Unity Pro editor.
  This choice requires the application program to be downloaded.
- Using the Command Register system bit `%SW60.8`.
  This choice must be performed online in the Primary CPU.

Selecting/deselecting address swap at Switchover:

Using Hot Standby Menu in Editor	Using the Command Register system bit `%SW60.8`
1. Open Hot Standby menu in Unity Pro. **2.** Go to Swap Address at Switchover area. **3.** Deselect Modbus Port 1. **4.** Verify modifications. **5.** Download application program to controller. **6.** Perform Switchover. **7.** Ensure Standby CPU switch to Primary CPU. **8.** Perform application program transfer.	**1.** Connect to Primary CPU. **2.** Access the Command Register System bit `%SW60.8`. **3.** Set the bit to 1. The default is 0.

<table>
<tr><th>Using Hot Standby Menu in Editor</th><th>Using the Command Register system bit <code>%SW60.8</code></th></tr>
<tr><td colspan="2">When a Switchover occurs:
If you change the options, the port addresses are not affected until Switchover occurs.
If NOM modules are used in the configuration, the offset of the Modbus address is +/-32 after the Modbus Plus address Switchover.</td></tr>
</table>

- Switchover with swapping Modbus addresses
  If controller A is the Primary CPU controller and its Modbus port has an address of 1, then the default addresses for the comparable port on controller B, the Standby CPU, is 129, which is 1 plus the offset of 128.
  If controller B becomes the Primary CPU controller as the result of a Switchover, its Modbus port assumes the address of 1, and the comparable port on controller A assumes the address of 129.
- Switchover without swapping Modbus addresses
  If controller A is the Primary CPU controller and its Modbus port 1 address is 1, then that port address remains at 1 after the switchover occurs. Likewise, if controller B becomes the Primary CPU controller as a result of a switchover, its Modbus port 1 address is remains at 1.

Swapping Modbus Plus Addresses at Switchover

In a Quantum Hot Standby system, the Modbus Plus port addresses on the Standby CPU controller are offset +/-32 from the comparable ports on the Primary CPU controller.

Modbus Plus address swap behavior at Switchover:

<table>
<tr><td>Default Behavior before Switchover:
Controller A = Primary CPU
MB+ address = 1
Controller B = Standby CPU
MB+ address = 33 (1 +32)
(+32 = Offset)</td></tr>
<tr><td>After switchover occurs:
Controller A = new Standby CPU
MB+ address = 33 (1 +32)
Controller B = new Primary CPU
MB+ address = 1</td></tr>
<tr><td>NOTE: Numerical address of both ports (A and B) range: 1 - 64.
If Primary CPU address = 50, corresponding Standby CPU = 18 (50 - 32)</td></tr>
</table>

The Modbus Plus address of the controllers can be changed using the front panel keypad: **Communication →Modbus Plus →Modify Address**

NOTE: The Modbus Plus port will be inactive for about 10 seconds after the RUN standby CPU state is shown on the LCD display.

Modbus Plus address swap behavior when the address is changed:

Forced behavior before Switchover: • Controller A = Primary CPU MB+ address = 1 • Controller B = Standby CPU MB+ address = 33 (1 + 32) (+32 = Offset)
Change address of Primary CPU = 5: • Controller A = Primary CPU MB+ address = 5 • Controller B = Standby CPU MB+ address = 33
Transfer Application Program: • Controller A = Primary CPU MB+ address = 5 • Controller B = Standby CPU MB+ address = 37 (5 + 32)
Force Switchover: • Controller A = new Standby CPU MB+ address = 37 (5 + 32) • Controller B = new Primary CPU MB+ address = 5
If the Modbus Plus address is modified, perform an Application Program Transfer. If not performed, a transfer creates a different offset address in the Standby CPU.

NOTE: At Switchover, the Quantum Hot Standby system and NOM modules swap Modbus Plus addresses almost instantaneously (within one or two milliseconds). This almost instantaneous Switchover means that host devices which are polling the controller must be talking to the Primary CPU controller and that the network should have minimal network interruption during Switchover.

NOTE: When using Modbus Plus communication and OSLoader, only address 1 is valid.

Swapping NOE IP Addresses at Switchover

When used in a Quantum Hot Standby system, the Quantum Ethernet TCP/IP network modules 140 NOE 771 01 and 11 support address swapping at Switchover. The swapping of IP addresses behaves much like the address swap of the Modbus Plus ports, except that the offset is 1 instead of 32.

At Switchover, the modules exchange their IP addresses. NOE 771 address swapping occurs automatically and can not be controlled by options selected in any of the tabs of the editor or controlled by turning ON/OFF any of the bits in the command register.

All standard rules apply to IP addressing with the additional restriction that the IP address cannot be greater than 253 or the broadcast address minus 2. Also, no other device should be assigned the IP address configured in Unity Pro +1.

NOTE:

NOE 771 01 and 11 address swap:
- NOE 771 01 and 11 modules are the only Ethernet option modules that support the IP address swap in Quantum Hot Standby with Unity Pro V2.0.
- NOE 771 01 and 11 modules must be configured in the same slot of the Primary CPU and Standby CPU backplanes.
- NOE 771 01and 11 modules require a minimum firmware revision 2.0 or higher.

Quantum Ethernet I/O IP Addresses at Switchover

The Quantum Ethernet I/O IP addresses are assigned to the CRP A and CRP B when the Hot Standby system is first configured and during a Switchover the addresses do not change.

Testing Switchover of a Quantum Hot Standby System

Testing Methods (First Time)

Follow these steps to conduct tests to observe:
- Hot Standby start-up
- automatic application program transfer
- Switchover of control from Primary CPU to Standby CPU

These tests are not necessary but helpful. If your racks are horizontally parallel and within 1 meter (3 feet) apart, the transfer process is easier to observe.

Hot Standby Start-up and Application Program Transfer

Follow these steps for start-up and application program transfer:

Step	Action
1	Configure two racks with identical hardware and firmware.
2	Connect to a Remote I/O (RIO) drop *(see page 83)*. **NOTE:** Ensure that the fiber optic Sync-link cable is connected between the controllers.
3	Start Unity Pro software and configure the local rack and the Remote I/O drop for your physical configuration.
4	Execute the `Build Project` command and save your application program.
5	Power up and connect Unity Pro to one controller. **NOTE:** The front panel keypad displays `No Conf`.
6	Download your application program and put the controller in the RUN mode. **NOTE:** The controller becomes the RUN Primary CPU.
7	Power up the other controller. **NOTE:** Application Program Transfer occurs automatically. The other controller becomes the RUN Standby CPU.
8	Ensure the Primary CPU and Standby CPU controllers are in RUN Primary CPU and RUN Standby CPU mode.

Preparing to Switchover

After completing the Hot Standby Start-up and Application Program Transfer *(see page 138)* procedure, your Quantum Hot Standby system is ready to perform a Switchover. Perform the Switchover using either:

- Hot Standby submenu on the front panel keypad
- Command Register, system bit `%SW60.1` or `%SW60.2`

NOTE: To observe a Switchover effect on the I/O modules, configure the Remote I/O (RIO) drop with a discrete output module during your initial start-up. Before performing a Switchover, connect to the Primary CPU and force the output bits in the module. Perform the Switchover and take note of the bumpless Switchover effect on the forced bits.

Switchover Test Using Front Panel Keypad

To force a Switchover using the front panel keypad, do the following:

Step	Action
1	Access the front panel keypad of the Primary CPU controller.
2	Go to **PLC Operation →Hot Standby →Hot Standby Mode**
3	Change Run to Offline. **NOTE:** Ensure that the Standby CPU switches to Primary CPU.
4	Change Offline to Run. **NOTE:** Ensure that the LCD displays `Run Standby CPU`.

Switchover Test Using Command Register

Follow these steps.

Step	Action
1	Connect Unity Pro to the Primary CPU.

2	Observe if the controller order on the Primary CPU is A or B using either of the following methods:
	• Unity Pro status dialog:
	Refer to the bottom of the Unity Pro window when connected online.
	• Access the Command Register system bits:
	  • If the connected Primary CPU is A, set.%SW60.1 to 0.
	  • If the connected Primary CPU is B, set.%SW60.2 to 0.
	NOTE: Ensure that the Standby CPU switched to Primary CPU
	If bits %SW60.1 and %SW60.2 are set to 0 simultaneously, a switchover occurs:
	• Primary PLC goes to RUN Offline
	• Standby PLC goes to RUN Primary

Step	Action
3	Connect Unity Pro to the new Primary CPU.
4	Access the Command Register system bits as in Step 2 and set them ti 1. **NOTE:** Ensure Standby CPU displays RUN Standby CPU. **NOTE:** Ensure the Primary CPU and Standby CPU controllers are in RUN Primary CPU and RUN Standby CPU mode.

Warm Start Restarting recommendation

After a global power loss while the system is running, the two CPUs synchronize each other at power up (primary PLC selection).

To insure synchronization at power recovery, two methods are proposed:

- The two PLCs must be powered at the same time (within 500 ms).

NOTE: In this case, the CPU with lower MAC address starts as Primary.

- The two PLCs must be powered one after the other with a minimum delay of 2 seconds.

NOTE: This second solution allows user to select which CPU becomes Primary (the first that is powered up).

4.2 EFBs for Quantum Hot Standby

Overview

This section describes the Quantum Hot Standby elementary function blocks (EFBs):
- HSBY_RD
- HSBY_ST
- HSBY_WR
- REV_XFER

What's in this Section?

This section contains the following topics:

HSBY_RD

Function Description

This EFB allows you to use the Hot Standby function. It searches (together with the other Hot Standby EFBs) the configuration of the respective Quantum PLC for the required components.

These components refer to hardware that is actually connected. Therefore, the correct behavior of this EFB on the simulators cannot be guaranteed.

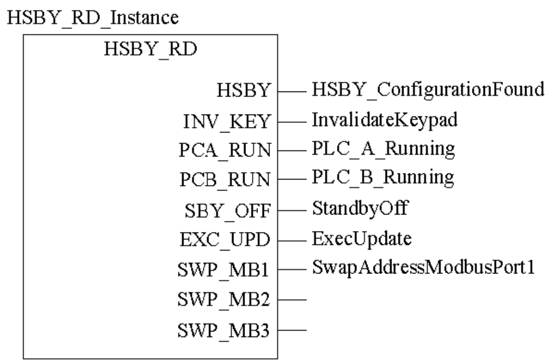
The **HSBY_RD** EFB checks System Word *(see Unity Pro, Program Languages and Structure, Reference Manual)* %SW60 to see if a Hot Standby configuration exists:

- If a Hot Standby configuration is present the contents of the command register are returned and the `HSBY_ConfigurationFound` output parameter is set to 1.
- If a Hot Standby configuration is not present the contents of the command register are returned and the `HSBY_ConfigurationFound` output parameter is set to 0.

`EN` and `ENO` can be configured as additional parameters.

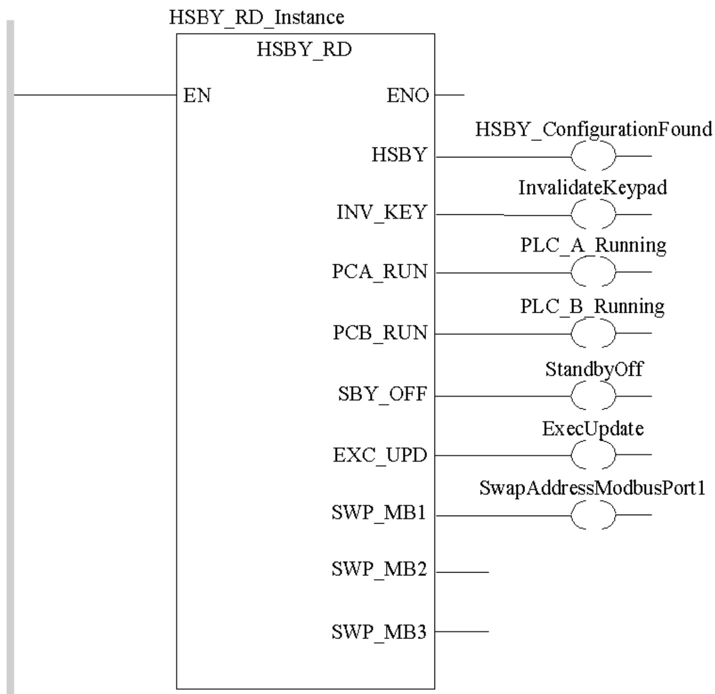
Representation in FBD

Representation:



Representation in LD

Representation:



Representation in IL

Representation:

```
CAL HSBY_RD_Instance (HSBY=>HSBY_ConfigurationFound,
        INV_KEY=>InvalidateKeypad, PCA_RUN=>PLC_A_Running,
        PCB_RUN=>PLC_B_Running, SBY_OFF=>StandbyOff,
        EXC_UPD=>ExecUpdate, SWP_MB1=>SwapAddressModbusPort1)
```

Representation in ST

Representation:

```
HSBY_RD_Instance (HSBY=>HSBY_ConfigurationFound,
        INV_KEY=>InvalidateKeypad, PCA_RUN=>PLC_A_Running,
        PCB_RUN=>PLC_B_Running, SBY_OFF=>StandbyOff,
        EXC_UPD=>ExecUpdate,
        SWP_MB1=>SwapAddressModbusPort1);
```

Parameter Descriptions

Description of the output parameters:

Parameter	Data type	Meaning
HSBY	BOOL	1 = Hot Standby configuration found 0 = Hot Standby configuration not found
INV_KEY	BOOL	1 = The submenu for the Hot Standby PLC button is disabled. 0 = The submenu for the Hot Standby PLC button is not disabled.
PCA_RUN	BOOL	For the local rack PLC with the Hot Standby CPU A: 1 = Command Register is selected for RUN 0 = Command Register is selected for OFFLINE
PCB_RUN	BOOL	For the local rack PLC with the Hot Standby CPU B: 1 = Command Register is selected for RUN 0 = Command Register is selected for OFFLINE
SBY_OFF	BOOL	1 = ??? 0 = The Standby PLC switches to the OFFLINE mode as soon as both PLCs receive a different program.
EXC_UPD	BOOL	1 = Operating System update in the Standby-PLC is possible with the primary CPU PLC still running. 0 = ??? (After Operating System Update, the Standby CPU PLC changes back to the ONLINE mode.)
SWP_MB1	BOOL	If a switchover has occurred, for Modbus ports 1: 1 = There is no swapping of addresses 0 = There is swapping of address
SWP_MB2	BOOL	Not used. Reserved
SWP_MB3	BOOL	Not used. Reserved

HSBY_ST

Function Description

This EFB allows you to use the Hot Standby function. It searches (together with the other Hot Standby EFBs) the configuration of the respective Quantum PLC for the required components.

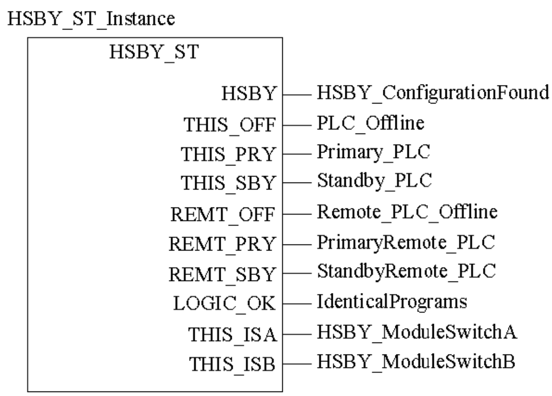
These components refer to hardware that is actually connected. Therefore, the correct behavior of this EFB on the simulators cannot be guaranteed.

This EFB is used to read the IEC Hot Standby status register (`%SW61` *(see Unity Pro, Program Languages and Structure, Reference Manual)*). If there is no Hot Standby configuration present, the `HSBY` output is set to 0.

`EN` and `ENO` can be configured as additional parameters.

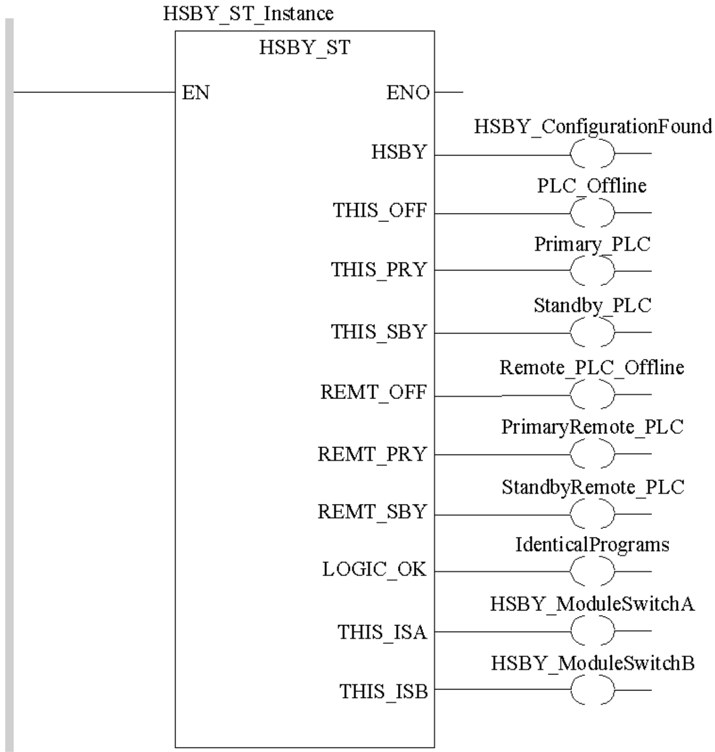
Representation in FBD

Representation:



Representation in LD

Representation:



Representation in IL

Representation:

```
CAL HSBY_ST_Instance (HSBY=>HSBY_ConfigurationFound,
         THIS_OFF=>PLC_Offline, THIS_PRY=>Primary_PLC,
         THIS_SBY=>Standby_PLC,
         REMT_OFF=>Remote_PLC_Offline,
         REMT_PRY=>PrimaryRemote_PLC,
         REMT_SBY=>StandbyRemote_PLC,
         LOGIC_OK=>IdenticalPrograms,
         THIS_ISA=>HSBY_ModuleSwitchA,
         THIS_ISB=>HSBY_ModuleSwitchB)
```

Representation in ST

Representation:

```
HSBY_ST_Instance (HSBY=>HSBY_ConfigurationFound,
         THIS_OFF=>PLC_Offline, THIS_PRY=>Primary_PLC,
         THIS_SBY=>Standby_PLC,
         REMT_OFF=>Remote_PLC_Offline,
         REMT_PRY=>PrimaryRemote_PLC,
         REMT_SBY=>StandbyRemote_PLC,
         LOGIC_OK=>IdenticalPrograms,
         THIS_ISA=>HSBY_ModuleSwitchA,
         THIS_ISB=>HSBY_ModuleSwitchB);
```

Parameter Descriptions

Description of output parameters:

Parameter	Data type	Meaning
HSBY	BOOL	1 = Hot Standby configuration found 0 = Hot Standby configuration not found
THIS_OFF	BOOL	1 = This PLC is offline 0 = This PLC is not offline
THIS_PRY	BOOL	1 = This PLC is the Primary CPU PLC 0 = This PLC is not the Primary CPU PLC
THIS_SBY	BOOL	1 = This PLC is the Standby CPU PLC 0 = This PLC is not the Standby CPU PLC
REMT_OFF	BOOL	1 = The other (remote) PLC is OFFLINE 0 = The other (remote) PLC is not OFFLINE
REMT_PRY	BOOL	1 = The other PLC is the Primary CPU PLC 0 = The other PLC is not the Primary CPU PLC
REMT_SBY	BOOL	1 = The other PLC is the Standby CPU PLC 0 = The other PLC is not the Standby CPU PLC
LOGIC_OK	BOOL	1 = The programs for both PLCs are identical and application mismatch is active. 0 = The programs are not the identical.
THIS_ISA	BOOL	1 = This PLC chose the CPU with the lower IP address between both Hot Standby CPUs. This is the Hot Standby CPU A. 0 = This is not CPU A.
THIS_ISB	BOOL	1 = This PLC chose the CPU with the higher IP address between both Hot Standby CPUs. This is the Hot Standby CPU B. 0 = This is not CPU B.

HSBY_WR

Function Description

This EFB allows you to use the Hot Standby function. It searches (together with the other Hot Standby Fibs) the configuration of the respective Quantum PLC for the required components.

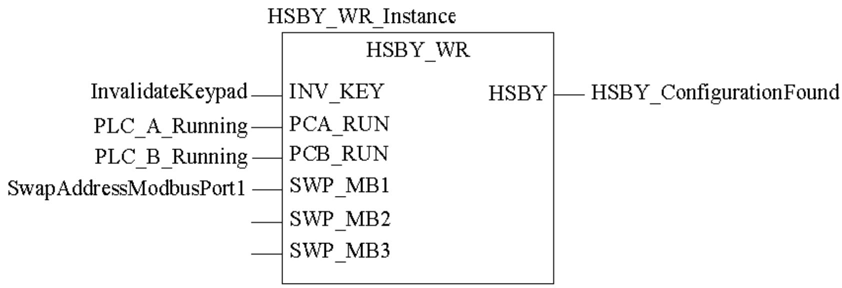
These components refer to hardware that is actually connected. Therefore, the correct behavior of this EFB on the simulators cannot be guaranteed.

`HSBY_WR` is used to set different Hot Standby Modes for the Primary CPU. Setting the respective modes means changing the Hot Standby Command register (`%SW60` *(see Unity Pro, Program Languages and Structure, Reference Manual)*), which is carried out automatically by the function block. If there is no Hot Standby configuration, the `HSBY_ConfigurationFound` output is set to 0, otherwise it is set to 1.

`EN` and `ENO` can be configured as additional parameters.

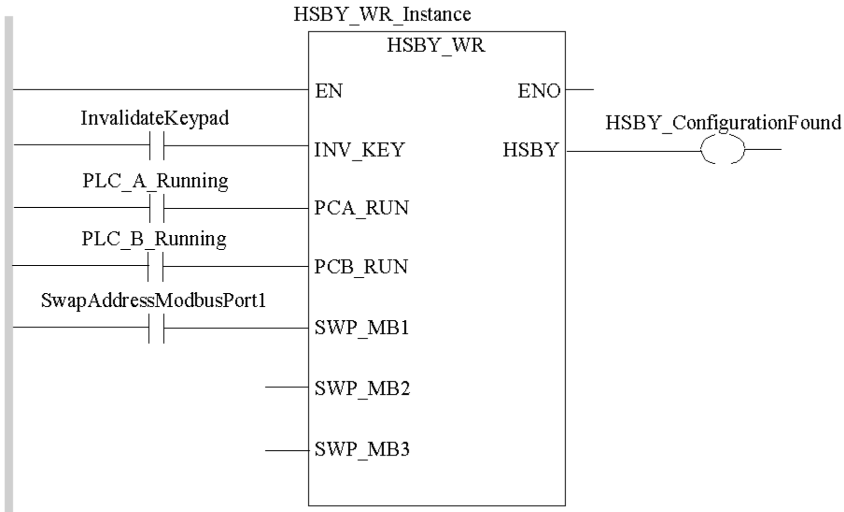
Representation in FBD

Representation:



Representation in LD

Representation:



Representation in IL

Representation:

```
CAL HSBY_WR_Instance (INV_KEY:=InvalidateKeypad,
         PCA_RUN:=PLC_A_Running, PCB_RUN:=PLC_B_Running,
         SWP_MB1:=SwapAddressModbusPort1,
         HSBY=>HSBY_ConfigurationFound)
```

Representation in ST

Representation:

```
HSBY_WR_Instance (INV_KEY:=InvalidateKeypad,
         PCA_RUN:=PLC_A_Running, PCB_RUN:=PLC_B_Running,
         SWP_MB1:=SwapAddressModbusPort1,
         HSBY=>HSBY_ConfigurationFound);
```

Parameter Description

Description of the input parameters:

Parameter	Data type	Meaning
INV_KEY	BOOL	In the submenu for the Hot Standby PLC button: 1 = Changes are not allowed. 0 = Changes are allowed.
PCA_RUN	BOOL	If 1 -> 0, then the Hot Standby 'A' CPU on the local rack is forced into OFFLINE mode. If 0 -> 1 and its button mode is in RUN mode, then the Hot Standby 'A' CPU is forced into the RUN mode.
PCB_RUN	BOOL	If 1 -> 0, then the Hot Standby 'B' CPU on the local rack is forced into OFFLINE mode. If 0 -> 1 and its button mode is in RUN mode, then the Hot Standby 'B' CPU is forced into the RUN mode.
SWP_MB1	BOOL	If 0 and there is a Switchover, then the Modbus address on port 1 of the NEW Primary CPU PLC changes: ● New Primary CPU PLC address = old Primary CPU address ● New Standby CPU PLC address = new Primary CPU address + 128.
		If 1 and there is a Switchover, then the Modbus address on Port 1 of the PLC does not change: ● New Primary CPU PLC address = old Standby CPU address ● New Standby CPU PLC address = old Primary CPU address
SWP_MB2	BOOL	Not used. Reserved
SWP_MB3	BOOL	Not used. Reserved

Description of the output parameters:

Parameter	Data type	Meaning
HSBY	BOOL	1 = Hot Standby configuration found. 0 = Hot Standby configuration not found.

REV_XFER

Function Description

This EFB allows you to use the Hot Standby function. It searches (together with the other Hot Standby EFBs) the configuration of the respective Quantum PLCs for the required components. These components refer to hardware that is actually connected.

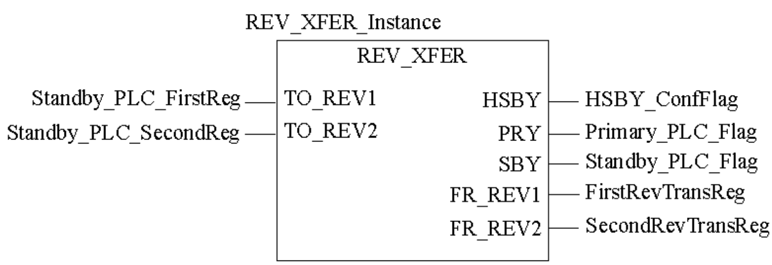
REV_XFER provides the ability to transmit 2 registers (%SW62/63) from the Standby PLC to the Primary PLC. The two registers EFB are used by the application program (in the first section) to register diagnostic information.

REV_XFER can only be used in the first executable section of the project. The parameter addresses TO_REV1 and TO_REV2 have to be in the Non-Transfer Area to prevent being overwritten by the Primary CPU.

As additional parameters, EN and ENO are projected.

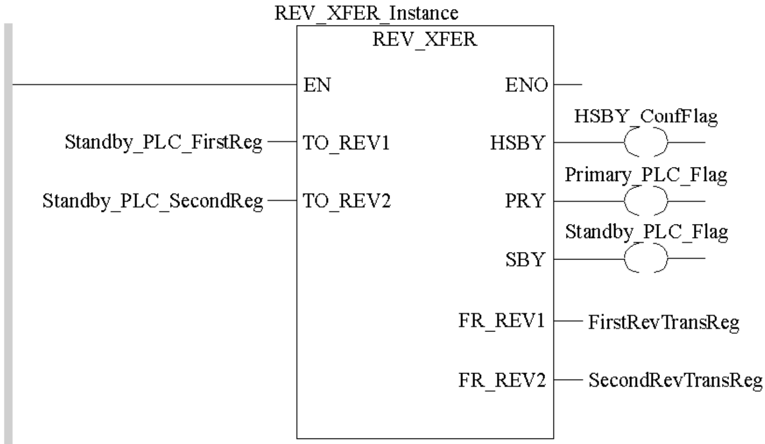
Appearance in FBD

Appearance:



Appearance in LD

Appearance:



Appearance in IL

Appearance:

```
CAL REV_XFER_Instance (TO_REV1:=Standby_PLC_FirstReg,
         TO_REV2:=Standby_PLC_SecondReg, HSBY=>HSBY_ConfFlag,
         PRY=>Primary_PLC_Flag, SBY=>Standby_PLC_Flag,
         FR_REV1=>FirstRevTransReg,
         FR_REV2=>SecondtRevTransReg)
```

Appearance in ST

Appearance:

```
REV_XFER_Instance (TO_REV1:=Standby_PLC_FirstReg,
         TO_REV2:=Standby_PLC_SecondReg, HSBY=>HSBY_ConfFlag,
         PRY=>Primary_PLC_Flag, SBY=>Standby_PLC_Flag,
         FR_REV1=>FirstRevTransReg,
         FR_REV2=>SecondtRevTransReg);
```

Parameter Description

Description of input parameters:

Parameter	Data type	Description
TO_REV1	INT	Describes the first reverse transfer register if this PLC is the Standby PLC. Data in this register are transferred from the Standby CPU to the Primary CPU at each scan.
TO_REV2	INT	Describes the second reverse transfer register if this PLC is the Standby CPU. Data in this register are transferred from the Standby CPU to the Primary CPU at each scan.

Description of the output parameters:

Parameter	Data type	Description
HSBY	BOOL	1 = This is a Hot Standby configuration. 0 = This is not a Hot Standby configuration.
PRY	BOOL	1 = This PLC is the Primary CPU PLC. 0 = This PLC is not the Primary CPU PLC.
SBY	BOOL	1 = This PLC is the Standby CPU PLC. 1 = This PLC is not the Standby CPU PLC.
FR_REV1	INT	Content of first reverse transfer register (%SW62 *(see Unity Pro, Program Languages and Structure, Reference Manual)*). Output only if HSBY is 1.
FR_REV2	INT	Content of second reverse transfer register (%SW63 *(see Unity Pro, Program Languages and Structure, Reference Manual)*). Output only if HSBY is 1.

4.3 Equipment Restrictions

Overview

This section describes equipment and application restrictions in a Quantum Hot Standby system.

What's in this Section?

This section contains the following topics:

Local and Distributed I/O Restrictions

Overview

A Quantum Hot Standby has the following I/O restrictions:

- In an Quantum Hot Standby system both Local I/O and Distributed I/O (DIO) can be used, they are not part of the redundant system.
- Local output can be dedicated to and managed by each PLC in their local racks.
- When local or Distributed I/O are used, they have to be managed in the first section of the MAST task of the application by using located %MW that are not transferred from the Primary CPU to the Standby CPU.
- Distributed I/O are not compatible with the safety processor (140 CPU 671 60S).

Local I/O Management

It is possible to manage outputs locally in both PLCs. They may be written with different values at the same time that depends on application program processing. For that, the first section of the MAST task of the application must be used. On the other hand, only the located variables that are not transferred from the Primary to the Standby must be used for managing the different values applied on the output modules.

When outputs are managed locally in each PLC, the output values must be evaluated in the first section of the MAST task at each PLC scan. If this is not done, the Standby output value is erased by the value coming from the Primary PLC.

CAUTION
RISK OF EQUIPMENT DAMAGE
Output values must be evaluated in the first section of the MAST task at each scan.
Failure to follow these instructions can result in equipment damage.

Handling I/O

The Quantum Hot Standby system supports I/O connected to a RIO drops and DIO connected using I/O scanning.

Local I/O can be configured and run, but is not redundant in a Quantum Hot Standby system.

Local I/O and PLC Modes

Local I/O is treated differently according to the operating mode of its PLC:

- Primary RUN
  The Local I/Os are updated by the application running in the Primary CPU and is exchanged with Standby CPU.
- Standby RUN
  The Local I/Os are updated by the application running in the Standby CPU.
- Run OFFLINE

  - All sections of the MAST task are executed
  - Only the first section of the MAST task is executed
  - No sections of the MAST task are executed

NOTE: The system always updates the Local I/O in Run OFFLINE.

NOTE: No data are transferred from the primary CPU to the OFFLINE CPU.

Module Restrictions

General

The Quantum Hot Standby with Unity Pro V2.0 and later versions do not support the following modules.

- 140 NOE 311 00
- 140 NOE 351 00
- 140 CHS 110 00
- 140 NOA 611 10
- 140 NOA 622 00
- 140 NOL 911 10
- 140 HLI 340 00

Application Restrictions

Timer Events and I/O Errors

Timer events are NOT synchronized in Quantum Hot Standby applications. Schneider Electric recommends not using timer events.

NOTE: If timer events are used, the detected I/O errors are not exchanged between Primary CPU and Standby CPU.

Mast Task Cycle Time and Watchdog

The Quantum Hot Standby system is optimized for applications with MAST task cycle times between,:

- for a s908 system, 30 ms and 250 ms
- for a Quantum Ethernet I/O Ethernet system, 30 ms and 350 ms

⚠ WARNING
UNEXPECTED EQUIPMENT OPERATION
The Drop hold up time must be set to at least 4 times the MAST task watchdog value.
Failure to follow these instructions can result in death, serious injury, or equipment damage.

4.4 PLC Communications

Overview

This section describes data and application transfers and the scan time.

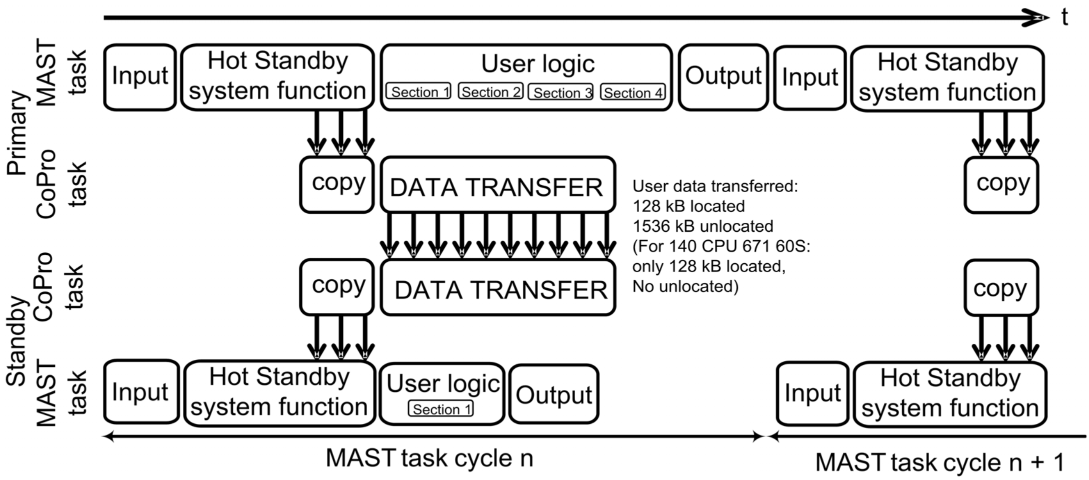
What's in this Section?

This section contains the following topics:

Data Transfer

Hot Standby Transfer Diagram

The following diagram illustrates the transfer of data from the Primary CPU to the Standby CPU Copro in a configuration using 140 CPU 67• •• processors:



Application Program Transfer

Overview

The Application Program Transfer (APT) feature provides you with the ability to configure the Standby CPU from the Primary CPU controller.

Use this feature to reprogram the Primary CPU controller or replace the Standby CPU controller because the process copies the full application program of the Primary CPU to the Standby CPU. This feature not only saves time but ensures that the controllers have identical configurations.

The system transfers the application program over the dedicated Quantum Hot Standby communications link between the two Copros.

Methods of Transferring Programs

Application transfer is from the Primary CPU to the Standby CPU.

The methods of transferring application programs are:

- Hot Standby submenu on the front panel keypad. *(see page 161)*
  Use either the Primary CPU or Standby CPU.
- Command Register system bit %SW60.5 *(see page 162)*.
  An application program transfer can be performed at any time.
- automatic transfer *(see page 162)* that occurs when you start a Hot Standby system for the first time. The Primary CPU automatically transfers the application program to the Standby CPU.
- select a Unity Pro command *(see page 163)*

NOTE: During application program transfer, the system is no longer redundant.

NOTE: If the Primary CPU stops before the other CPU is ready to assume the role of Primary CPU, the process is no longer controlled.

LCD Keypad Messages

During an APT the following messages may be displayed on the 2 CPU LCDs:

- Primary CPU:
  - Transferring
  - End of Transfer
- Standby CPU:
  - Program transfer in progress
  - Transferring
  - Transfer retry please wait
  - Transfer OK
  - Transfer NOK
  - Can’t transfer PLC reserved

Validating Transfer

The secondary CPU validates the transferred application program. After validating it starts automatically as the Standby CPU.

Transfer Time

The Application Program Transfer time depends on the size of the application program (the larger the program, the longer the time) and the type of MAST scan time:
- for a periodic MAST, the scan time is not effected by an APT
- for a cyclic MAST, the scan time may change during an APT

Updating from the Primary CPU

An application program update may only be performed from the Primary CPU to the Standby CPU.

NOTE: The Standby CPU controller cannot update the Primary CPU.

Transfer Size Limits

In the Quantum Hot Standby 140 CPU 67• •• transfer size depends on the configuration. For example, using a card bridge you may transfer up to 7 Mb.

Therefore, transfer the complete application program regardless of the size. This transfer takes place over multiple scans and is broken up into multiple transfer packets.

Application Program Transfer Using the Keypad

To transfer, use the front panel keypad on the controller unit (Primary CPU or Standby CPU). The Primary CPU copies the complete application program and data to the Standby CPU.

The following table shows the Application Program Transfer procedure.

Step	Action
1	Ensure the Primary CPU Controller is in RUN PRIMARY CPU mode. **Result:** The LCD on the PLC displays the mode as RUN PRIMARY CPU.
2	Check that both: ● invalidate Keypad option is NOT selected ● the key switch is unlocked

Step	Action
3	Go to the submenu **Hot Standby →Transfer**.
4	Push Enter to execute the application program transfer from the Primary CPU to the Standby CPU.

NOTE: The **Hot Standby →Transfer** command can be performed either in the Primary CPU or Standby CPU controller, but only the Standby CPU controller is updated.

Transferring the Application Program Using Command Register System Bit %SW60.5

To transfer, use the command register in the Unity Pro software. The Primary CPU copies the complete application program and data to the Standby CPU.

To transfer an application program (logic program or project) to either the Primary CPU or Standby CPU controller using Command Register system bit %SW60/5, do the following:

Step	Action
1	Connect to the Primary CPU or Standby CPU controller.
2	Access Command Register system bit %SW60.5.
3	Set bit to 1. Note: The process of setting the bit toggles the bit from 0 to 1 and back to 0.

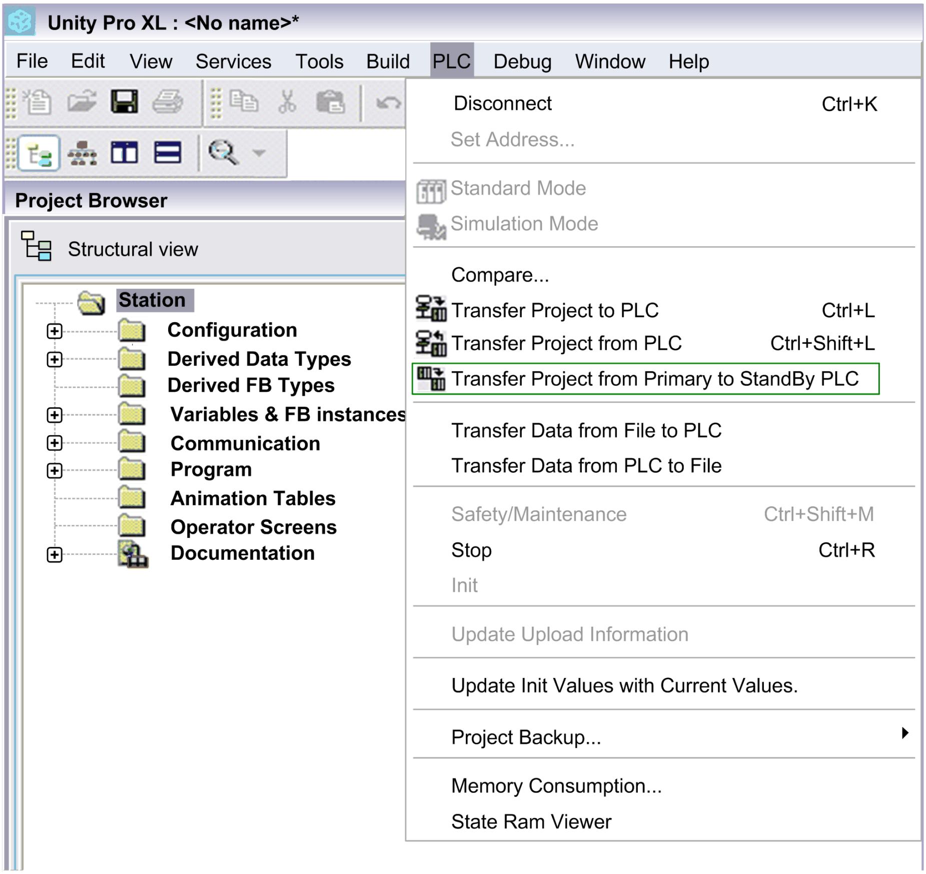
Automatic Application Program Transfer

As soon as a Primary CPU controller detects a blank controller, the Primary CPU transfers the program to the blank controller, which becomes the Standby CPU. After application program transfer both controllers have identical application programs.

NOTE: The controllers need to have the same configuration (with the same or no PCMCIA cards).

Unity Pro ATP Command

If Unity Pro is connected to the Primary PLC, it can launch an APT from the **PLC** menu:



If Unity Pro is connected to the Standby PLC, this menu item is not available.

Identical Configurations and Application Programs

After the transfer, the Primary CPU and Standby CPU have identical configurations and application programs.

In the event of a detected error in the Primary CPU and depending on the mode selected for the Standby CPU (Run or Offline), the Standby CPU may or may not be ready to assume the role of Primary CPU.

Scan Time

Effect on System Scan Time

The scan time of any Quantum Hot Standby system depends on the amount of data transferred.

Because data must be transferred from Primary CPU to Standby CPU, any Quantum Hot Standby system has a higher scan time than a comparable standalone system.

NOTE: In a Quantum Hot Standby system these 2 processors work in parallel:

- CPU performs application program processing
- Copro performs communication transfer

This reduces transfer times between the PLC and Unity Pro.

NOTE: Do not set the period of periodic MAST task below 30 ms.

Difference between CPU 671 60 and 60S modules

For the 140 CPU 671 60S module, the scan time and the figures are similar but the transferred data are different. There is no unlocated data. They are replaced by private data (data internally used by the application and not accessible for the user).

Performance Considerations in 140 CPU 67• ••

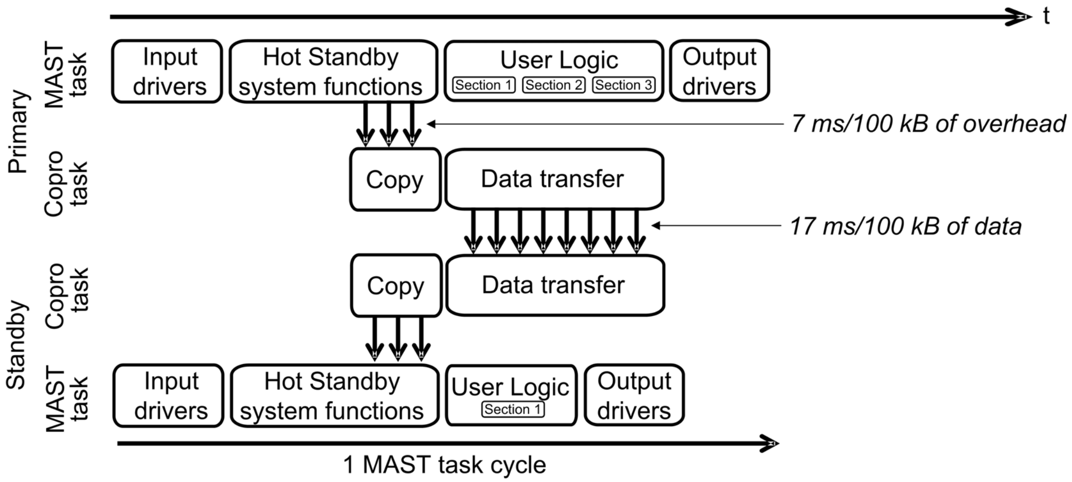
A Quantum Hot Standby system increases the length of a MAST scan, creating system overhead.

NOTE: System overhead is the time required to copy the application data to the communication link layer.

The network scan (communication between Primary CPU and Standby CPU Copros):

- exchanges data between both controllers
- runs in parallel with the application program

A Hot Standby system with a 140 CPU 67• •• is illustrated below:



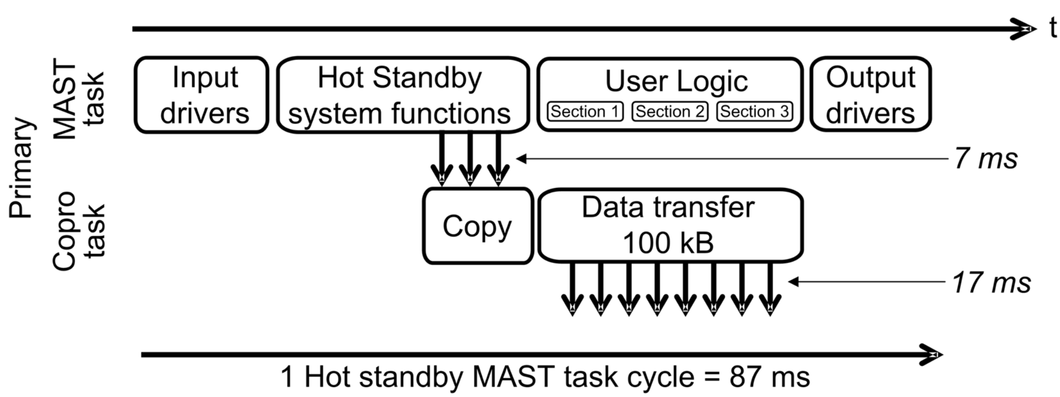
Most of time, the MAST scan is greater than the network scan.

However, when processing some application programs, additional system overhead may occur.

Examples 1

For this example:
- Standalone application scan time: 80 ms
- data transferred (state RAM + unlocated variables): 100 kB

The Hot Standby MAST cycle is greater than the Standalone scan time only by the Hot Standby 7 ms overhead.

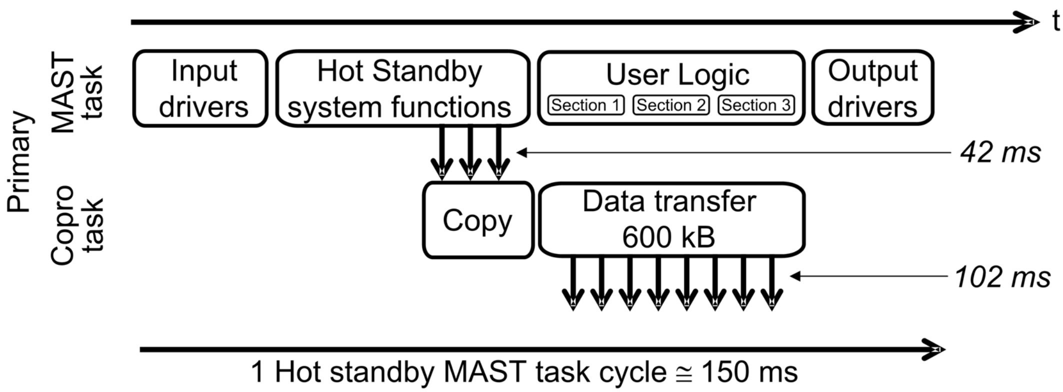


Example 2

For this example:

- the standalone application scan time: 80 ms
- data transferred (state RAM + unlocated variables): 600 kB

The Hot Standby MAST cycle is much greater than the Standalone scan time because of the long data transfer time.



4.5 Developing A Hot Standby Application

Purpose

This section describes the rules for developing an application for a Quantum Hot Standby system.

What's in this Section?

This section contains the following topics:

Adjusting MAST Task Properties

Introduction

After reviewing the MAST task execution modes, this topic describes adjusting the MAST task period and execution time measurement procedures.

⚠ WARNING

UNEXPECTED EQUIPMENT BEHAVIOR

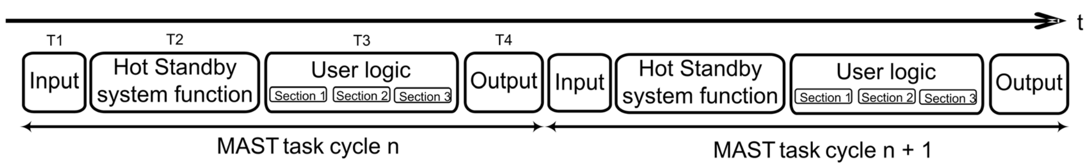
Design your application in such a way that your process is not impacted by a cycle time variation **that might appear after a firmware upgrade**.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

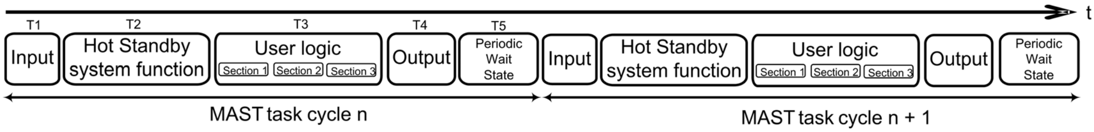
Review of MAST Task Execution Modes

The MAST task can be configured to use one of two execution modes:

- Cyclic mode:
  In the cyclic execution mode, MAST tasks occur in sequence with no reference to the system clock without any delay between tasks other than a very brief period of system processing. Almost immediately after one task completes another begins. Therefore, the actual duration of a MAST task in the cyclic mode can vary significantly depending on the size and activity of the application, and the number of inputs and outputs to be controlled.



- Periodic mode:
  In the periodic execution mode, MAST tasks are sequenced according to a countdown timer which is referenced to the system clock. This countdown timer may be set for a period between 1 and 255 ms. If the countdown expires before the end of the task, the task completes normally. If this occurs regularly, the system will appear as if the cyclic MAST task execution mode had been selected. However, some applications, such as process control, do require regular cycle times. If this is the case for your application, confirm that the task period is of sufficient length to avoid cyclic-like behavior.



Execution Time Measurement

The execution time of the MAST task can be measured by reading system words:
- %SW30: Execution time (in ms) of the last task
- %SW31: Execution time (in ms) of the longest task
- %SW32: Execution time (in ms) of the shortest task

In both cyclic and periodic mode, the MAST execution time is the sum of T1 + T2 + T3 + T4.

T5 of the periodic mode is not taken into account.

First Step of Execution Time Measurement

To measure the execution time of the MAST task in a Quantum Hot Standby configuration, it is advised to measure first the execution time in standalone mode (or with one of the two PLC in STOP) with the MAST task configured in cyclic mode. In this case, there is no data exchange between the two PLCs, and the execution time of the Hot Standby Copro part (T2) is reduced to its minimum.

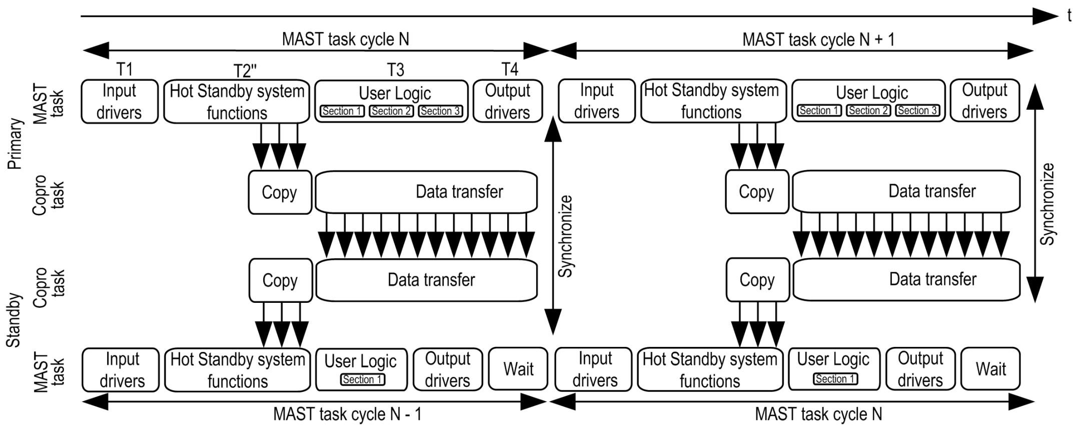
The execution time of the last MAST cycle = %SW30 = T1 + T2 + T3 + T4.

Second Step of Execution Time Measurement

In a second step, the execution time has to be measured with a Primary and Standby PLC.

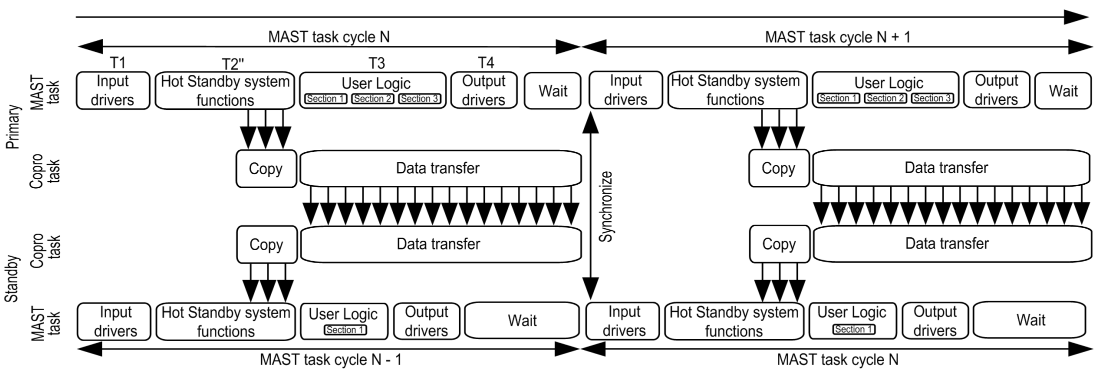
Two cases have to be taken into account:

1. The data transfer has no impact on the Primary MAST task duration:



In this case, the execution time of the Hot Standby part (T2’) is increased with the time required to copy the data base from the CPU memory to the Hot Standby Copro shared memory.
The execution time of the last MAST cycle = %SW30 = T1 + T2’ + T3 + T4 with T2’ = T2 + time to copy the data base from the CPU memory to the Copro shared memory.
There is no need for a Wait state to be added to the Primary MAST task because the User Logic “covers” the Copro data transfers.

2. The data transfer has an impact on the Primary MAST task duration:



In this case, the time to copy to the Copro shared memory, increases T2" compared to T2.

Also, because of the increased data, the data transfer between the Primary and Standby Copros requires a Wait state in the Primary MAST task.
The execution time of the last MAST cycle = %SW30
= T1 + T2'' + T3 + T4 + Wait state where T2'' = T2 + time to copy the data base from the CPU memory to the Copro + time to transmit all the data on the network and free the Copro shared memory.

Third Step of Execution Time Measurement

In the periodic mode, it appears that the execution time that is measured is lower than in the cyclic mode. In certain cases, the difference between the two execution modes can be large.

Procedure to Adjust MAST Task in Periodic Mode

If the MAST task has to be configured in the periodic mode, it is recommended to:

Step	Action
1	Measure the maximum value (%SW31) of the MAST task in cyclic mode with the Quantum Hot Standby Primary and Standby PLCs) running normally. This measure has to be done in the Primary PLC with all the configured tasks active (only the MAST task is recommended in a Quantum Hot Standby application).
2	Configure the periodic mode with a period at least equal to %SW31 plus a margin of around 20%, that is, Period = %SW31 + (%SW31 * 20%).

How to Program a Quantum Hot Standby Application

Processor Configuration

There are two types of MAST task execution modes:

- Cyclic - The MAST task executes as rapidly as possible.
- Periodic - The MAST task delays execution (if necessary) to observe a user-defined minimum cycle time.

When the periodic mode is used, the user-defined period must take into account the longer mast task periods necessary in a redundant system.

The following table presents the characteristics of MAST tasks that may be adjusted by the user in Unity Pro:

Characteristics	Unity Pro Default Values
Max period (ms)	255
Default period (ms)	20 (**Note:** Set to 80 ms as an initial value for Hot Standby systems)
Min. period (ms)	1 (0 if the Cyclic MAST task execution mode is selected)
Period increment (ms)	1
Max Watchdog (ms)	1500
Default Watchdog (ms)	250
Min. Watchdog (ms)	10

For more details, see Adjusting Mast Task Properties *(see page 168)*.

Detecting Cold and Warm Starts of a Quantum Hot Standby PLC

In a Quantum Hot Standby PLC, only the System Word %SW10 and the System Bit %S1 can be used to detect respectively a Cold Start and a Warm Start.

- **%SW10** (Cold-start test):
  If the value of the %SW10.0 bit (representing the MAST task) is set to 0, this means that the task is performing its first cycle after a Cold Start.
  At the end of the first cycle of the MAST task, the system sets the bit %SW10.0 to 1.

- **%S1** (Warm-start test):
  The default value of %S1 Is 0. This bit is set to 1 when the device power is cycled and a data save operation is performed. If this value is 1, this is an indication that the last start performed was a Warm Start.
  It is reset to 0 by the system at the end of the first complete cycle, but before the outputs are updated.

To process your application based on the type of start, the program must test whether %SW10.0 is reset to 0 (or %S1 is set to 1) at the start of first MAST task. %SW10 and %S1 can be tested by the application in either the Primary or Standby mode.

Transferring Your Program to the Primary and Standby PLCs

Transferring Your Program

Because a Hot Standby system requires that identical application programs exist on both the Primary and Standby PLCs, you must upload your application twice, once to each PLC.

The procedure is the same for both PLCs:

Step	Action
1	Connect the PC with Unity Pro (version 3.1 or above) to a USB port on the PLC
2	Use the Unity Pro command: **PLC →Transfer program to PLC**

NOTE: If your Hot Standby system is already configured and uses an Ethernet RIO head, you must stop all your system before downloading a new application (same recommandation after a rebuild all).

4.6 Debugging a Hot Standby Application

Debugging

Introduction

You can write an application for your Quantum Hot Standby system in almost the same manner as you would for any other Quantum PLC. This is because the Quantum Hot Standby system does not require the use of special function blocks or user actions to provide most redundant features. There are some important exceptions to this statement. See Restricted Functions *(see page 157)*

Debug and Diagnostic

The following tables presents Debug and Diagnostic operations for Quantum Hot Standby PLCs:

Diagnostic		**140 CPU 671 60**	**140 CPU 671 60S**	**140 CPU 672 61**
Diagnostic Function Block		Yes	Yes	Yes
Diagnostic Buffer		Yes	Yes	Yes
Diag. buffer characteristics	Max buffer size	16KB	25KB	25KB
	Max errors	160	254	254
Breakpoint*		1 max	1 max	1 max
Step by step (Into, over, and out)		Yes	Yes	Yes
Variable animation		• End of MAST • Watch Point	• End of MAST • Watch Point	• End of MAST • Watch Point
Link animation		Yes	Yes	Yes

* Do not use breakpoints on a Primary CPU as this will cause a Switchover.

Debug the Control/Command of the Procedure

Debugging a Quantum Hot Standby application program is a two-stage process:

1. Debug the basic program operation on one of the Hot Standby PLCs running as a Standalone PLC. When you do this, all of the debug and diagnostic resources noted in the tables above are available.

NOTE: If a standalone Hot Standby controller is not available, put the Standby PLC into a Non Conf state and do this first stage of debugging on the Primary PLC.

2. Debug any redundancy-specific aspects of your program on a functioning (redundant) Hot Standby system that is not actively managing your process. When performing this second stage, the debug and diagnostic resources in the tables above are not available.

⚠ WARNING

UNINTENDED EQUIPMENT OPERATION

When debugging the redundancy-specific aspects of your application program:

- Always debug your application on a fully functioning Hot Standby system.
- Only conduct debugging on a Hot Standby system that is not actively managing your process.
- Do not use the Unity Pro debug and diagnostic features except as permitted by this manual.
- Confirm that the interaction of the MAST task mode and duration with the Watchdog values meet the needs of your application.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

When you are conducting the second stage of debugging, confirm that you initially connect to the PLC currently acting as Primary. The Standby PLC only executes section 0 of the MAST task of your application program.

Debug the First Section of the MAST task in Standby PLC

To debug the first section in the Standby PLC application, the following points have to be taken into account:

- All the application data that comes from the Primary PLC are displayed in a Standby animation table.
- Animation tables can be synchronized with watch points. This is the best way to animate data in synchronization with code execution. Refer to the *Unity Pro Program Languages and Structure Reference Manual*, reference 35006144, for further details.

Debugging the Redundancy Part

Do not attempt to debug or otherwise verify the performance of an application intended for use in a Hot Standby system on a non-Hot Standby PLC. You must debug Hot Standby-related applications on a Hot Standby PLC.

Do not use the debug and diagnostic operations normally available for Quantum PLCs on a redundant Hot Standby system. The operations such as step-by-step and breakpoints halt program execution and eliminate the redundancy of the Hot Standby system.

NOTE: A Switchover is not generated when the Primary application stops on a breakpoint.

The debugging that can be accomplished once your application program is loaded onto a redundant Hot Standby system offers the following debugging features:

- Static verification. Check that:
  - application restrictions *(see page 157)* in this manual have been observed
  - MAST task characteristics have been configured properly
- Dynamic verification
  After each PLC has been made live (application already transferred), check that the redundancy function is correctly performed in each PLC: the Status register bit %SW61.15 is equal to 1 and the bit %SW61.6 is equal to 0.

Once the Hot Standby PLCs have entered either the Primary or Standby operating modes, confirm that:

- all application program sections of the MAST task are executed on the Primary PLC
- only the first section of the MAST task is executed in the Standby PLC

Modifying and Upgrading

III

Purpose

This part describes for a Quantum Hot Standby system:
- handling logic mismatch
- transferring application programs
- enabling an Operating System upgrade

What's in this Part?

This part contains the following chapters:

Application Modifications

5

Overview

This chapter provides information about making Quantum Hot Standby application modifications with Unity Pro.

What's in this Chapter?

This chapter contains the following topics:

Quantum Hot Standby Application Mismatches

Identical Application Programs

In a fault-tolerant redundant system and under normal operating conditions, both controllers must load the identical application programs). The application program is updated every scan by transferring data from the Primary CPU to the Standby CPU. Both controllers conduct tests to detect if a mismatch exists between the application programs.

Differences between the following conditions cause a mismatch in the application program:

- programs
- animation tables
- comments (on variables)
- I/O configuration changes in the RUN mode

NOTE: To exclude animation tables and comments (on variables) from an uploaded application without causing a mismatch

- select **Tools** →**Project Settings** →**Build tabs** (default).
- in the Upload Information area select `without`

When a mismatch exists, a Switchover is not possible, and the Standby CPU controller does not go online. However, there are situations when you may want to allow a mismatch between the application programs. To enable this condition, use the Quantum Hot Standby Application Mismatch feature.

NOTE: Switchover cannot occur while the Standby CPU controller is offline.

Application Mismatch Definition

Application Mismatch is a Quantum Hot Standby feature that allows a mismatch between the application programs and the I/O configuration of the Primary CPU and Standby CPU.

Use the Application Mismatch feature to modify an application program and the I/O configuration while the process remains redundant.

Build Project Function

Use the Build Project function to perform an application mismatch with Unity Pro.

NOTE: Schneider Electric recommends that the **Rebuild All Project** not be used to create an application mismatch. The **Rebuild All Project** function creates a completely new project even if nothing has been changed in the application.

Causing a Mismatch

In the Quantum Hot Standby system, all memory is allocated by a memory manager, which automatically maps the logical memory to physical memory locations.

This dynamic data memory layout is the heart of the programming flexibility and platform independence that Unity Pro provides, but on a Quantum Hot Standby system with different user logic, dynamic data memory layout makes a cyclical data update very difficult. Therefore, mismatches occur.

Allowing a Mismatch

An application mismatch, without stopping the process controlled by the application program, allows the following:

- modification (edit) online of an application program in the Standby CPU while the Primary CPU controls the process *(see page 188)*
- modification (edit) online of an application program in the Primary CPU while the Primary CPU controls the process *(see page 189)*
- download an offline-modified application program to the Standby CPU and perform a Switchover to run the modified application program
- perform a CCOTF modification of the I/O configuration on the Primary PLC

Creating a Mismatch

Use one of these methods to create an application mismatch condition:

1. select **Online** in the **Standby CPU On Application Mismatch** group in the Unity Pro Hot Standby Tab dialog.
   This action requires the application program to be downloaded to the PLC.
2. set the Command Register system bit %SW60.3 to 1.
   This action must be performed online in the Primary CPU.

Transferring User Data during a Mismatch

The following table shows which user data is transferred when a mismatch occurs:

Data Type	Transferred on Application Mismatch
Located variables (State RAM)	Yes
Unlocated global variables	Yes (not for the 140 CPU 671 60S CPU), unless variables exist ONLY in modified controller
DFB & EFB instance data	Yes, unless data exist ONLY in modified controller
SFC variable area	Yes (not for the 140 CPU 671 60S CPU), unless associated SFC section is modified, refer to Modifying an SFC Section *(see page 130)*
System Bits and Words	Yes

Using Application Mismatch with Care

Ensure that there is no mismatch between I/O maps or configurations.

⚠ WARNING

UNEXPECTED APPLICATION BEHAVIOR

Ensure that both:
- I/O maps are identical
- configurations are identical

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Selecting the **Standby CPU On Application Mismatch** option, allows overriding the default condition (Standby CPU going offline).

If you change the parameter in this field from Offline to Online, the Standby CPU remains online if an application mismatch is detected between the application programs of the Standby CPU and Primary CPU.

Updating Section Data in an Application Program

All data of a section is updated every scan only if the data in the Standby CPU is the same as in the Primary CPU.

If the sections are equal on the Primary CPU and the Standby CPU, the following section data is updated:

- internal states of Elementary Function Blocks (EFBs) used in the section, for example, Timers, Counters and PID
- all Derived Function Block (DFB) data blocks of each DFB instantiated in the section, including nested DFBs

Updating Global Data in an Application Program

With **Application Mismatch** enabled, the application program global data is updated with every scan. Global data that does not exist on both controllers is not updated.

The application program's updated global data includes both:
- all declared variables in the Variable Editor
- all section and transition variables

The process of updating the application program global data in a Hot Standby system affects:
- declared variables
  All declared variables are updated on every scan if they are declared on both controllers.

- updating Standby CPU
  If a complete application program transfer is done to the controller that did not receive the modified changes, then both controllers have equal application programs and the Standby CPU controller is fully updated.
- deleted and re-declared variables
  If, due to a modification, a global variable has been deleted first, and then **re-declared**, this variable would be treated as a new variable, even if the same name is used. The update procedure must then be followed to bring the controllers to the same state.

NOTE: The system reserves space for these variables whether they are used in the controller application program or not.

Unused variables consume space and require time to be transferred from the Primary CPU to the Standby CPU. Therefore, in the Primary CPU application program, Schneider Electric does not recommend using variables that are defined but not used.

Online or Offline Modifications and Application Mismatch

Modifying Application Programs

Normally, once a fault-tolerant redundant system is configured, programmed, and controlling its process, the system is not shut down, not even for periodic maintenance. However, there may be situations when you may need to make modifications to the application program and continue to control the process.

The Application Mismatch feature allows you to modify application programs online or offline while controlling the process.

⚠ WARNING

UNEXPECTED EQUIPMENT BEHAVIOR

Before transferring a modified application to the Standby CPU:
- Examine carefully all the impacts of the modifications on the application.
- Check that the modified application does not have adverse effects on the process.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Standby CPU Online Application Modifications with Application Mismatch

Procedure

To make online modifications to an application program in the Standby PLC, follow these steps:

Step	Action
1	Verfiy that the Primary and Standby PLCs are in Run Primary CPU and Run Standby CPU modes.
2	Connect Unity Pro to the Primary CPU.
3	Set the Command Register system bit %SW60.3 to 1.
4	Connect Unity Pro to the Standby CPU controller.
5	Modify online the application program.
6	Perform a **Build Project**. **NOTE:** If adding/removing modules with CCOTF *(see Modicon Quantum, Change Configuration On The Fly, User Guide)*, use **Build Changes**.
7	Verfiy that the Primary and Standby PLCs are in Run Primary CPU and Run Standby CPU modes.
8	Perform a Switchover *(see page 190)*. Note: Standby CPU changes to Primary CPU.
9	Perform an application transfer *(see page 192)* to new Standby CPU.
10	Set the Command Register system bit %SW60.3 to 0. Note: The Command Register system bit is then returned to 0 from 1.

NOTE: For more details, refer to Application Mismatch *(see page 193)*.

Primary CPU Online Application Modifications with Allowed Application Mismatch

Procedure

To make online modifications to an application program in the Primary PLC, follow these steps:

Step	Action
1	Verify that the Primary and Standby PLCs are in Run Primary CPU and Run Standby CPU modes.
2	Connect Unity Pro to the Primary CPU.
3	Set the Command Register system bit %SW60.3 to 1
4	Modify online the application program.
5	Perform **Build Project**. **NOTE:** If adding/removing modules using CCOTF *(see Modicon Quantum, Change Configuration On The Fly, User Guide)*, use **Build Changes**.
6	Verify that Primary and Standby PLCs are in Run Primary CPU and Run Standby CPU modes.
7	Perform an application transfer *(see page 192)* to the Standby CPU.
8	Set the Command Register system bit %SW60.3 to 0. **NOTE:** The Command Register system bit is then returned to 0 from 1.

NOTE: For more details, refer to Application Mismatch *(see page 193)*.

Offline Application Modification with Allowed Application Mismatch

Procedure

To make offline modifications to an application program in either PLC, follow these steps:

Step	Action
1	Modify the application program offline.
2	Perform **Build Project** and save. **NOTE:** Do not use the **Rebuild All Project** option because this causes the Standby CPU to go offline when the application program is downloaded.
3	Verify that the Primary and Standby PLCs are in Run Primary CPU and Run Standby CPU modes.
4	Connect Unity Pro to the Primary CPU.
5	Set the Command Register system bit %SW60.3 to 1.
6	Connect Unity Pro to the Standby CPU and open the modified program.
7	Download the program and select RUN. **NOTE:** Check the controller state and ensure that it is in Run Standby mode.
8	Verify that the Primary and Standby PLCs are in Run Primary CPU and Run Standby CPU modes.
9	Perform a Switchover *(see page 190)*. **NOTE:** Ensure that the Standby CPU switched to the Primary CPU.
10	Perform an application transfer *(see page 192)* to the Standby CPU.
11	Set the Command Register system bit %SW60.3 to 0. **NOTE:** The Command Register system bit is then returned to 0 from 1.

NOTE: For more details, refer to Application Mismatch *(see page 193)*.

⚠ WARNING

UNEXPECTED EQUIPMENT BEHAVIOR

Before transferring a modified application to the Standby CPU:

- Examine carefully all the impacts of the modifications on the application.
- Check that the modified application does not have adverse effects on the process.

Failure to follow these instructions can result in death, serious injury, or equipment damage.

Switchover Methods with Application Mismatch

Ovewview

Switchover can be performed using one of two methods:
- Hot Standby submenu on the front panel keypad
- Change Command Register system bit %SW60.1 or %SW60.2

NOTE:

If bits %SW60.1 and %SW60.2 are set to 0 simultaneously, a Switchover occurs:
- Primary CPU controller goes RUN Offline, and
- Standby CPU controller now operates as RUN Primary CPU.

Switchover Using Front Panel Keypad

To force a Switchover using the front panel keypad, do the following:

Step	Action
1	Access the front panel keypad of the Primary CPU controller.
2	Go to **PLC Operation** menu.
3	Go to **Hot Standby** submenu.
4	Go to **Hot Standby** mode
5	Modify **Run** to **Offline**. Note: Verify that Standby CPU switched to Primary CPU.
6	Modify **offline** to **run**. Note: Verify that the LCD displays `Run Standby`.

Command Register Switchover

To perform the Switchover using Command Register system bit %SW60.1 or %SW60.2:
- save the application program twice under a different file names:
  - file 1
    Saved before modification
  - file 2
    Saved after modification
- verify the A/B order of the controller using one of two methods:
  - Hot Standby submenu on the front panel keypad **PLC Operation →Hot Standby →Hot Standby Order**.
  - Unity Pro status dialog (refer to the bottom of the Unity Pro window when connected online)

Switchover Using Command Register System Bit %SW60.1 or %SW60.2

To force a Switchover by setting the bits in the Command Register, do the following:

Step	Action
1	Open file 1 in Unity Pro.
2	Connect Unity Pro to the Primary CPU.
3	Verify the A/B controller order of the Primary CPU.
4	Set correct bit in %SW60: ● If the connected CPU is A, set Command Register bit %SW60.1 to 0. ● If the connected CPU is B, set Command Register bit %SW60.2 to 0. **NOTE:** Ensure that the Standby CPU switched to Primary CPU.
5	Open file 2.
6	Connect Unity Pro to the new Primary CPU controller.
7	Set Command Register system bit used in Step 4 to 1. **NOTE:** Verify that the Standby PLC is now online.
8	Verfify that both Primary and Standby CPUs are in Run Primary CPU and Run Standby CPU modes.

Manual Application Program Transfer Method and Application Mismatch

General

A manual Application Program Transfer can be performed using one of two methods:

- Hot Standby submenu on the front panel keypad
- Command Register system bit %SW60.5

Application Program Transfer Using Front Panel Keypad

To transfer an application program to either the Primary CPU or Standby CPU controller using the front panel keypad, do the following:

Step	Action
1	Access the front panel keypad of a controller (Primary or Standby)
2	Go to **PLC Operations** menu
3	Go to **Hot Standby** submenu
4	Go to Hot **Standby transfer** and press **ENTER** to confirm the transfer. **NOTE:** Verify that the transfer to Standby CPU occurs.

Application Program Transfer Using Command Register System Bit %SW60.5

To transfer an application program from the Primary CPU to the Standby CPU using Command Register system bit %SW60.5, do the following:

Step	Action
1	Connect unity Pro to the Primary CPU.
2	Set Command Register system bit %SW60.5 to 1. **NOTE:** This bit is reset 0 after the transfer.

Recommendations for Using Application Mismatch

General

When using the Application Mismatch feature the following are affected:

- Upload Information Management
- online modifications to the Standby CPU
- Application Program Transfer
- setting the Command Register system bit `%SW60.3`

Upload Information Management Feature

During online modifications, the system detects that the application-program information in the controller differs from the application-program information in the computer. Because this information is used later when an upload is performed, the system requires you to update this information and constantly presents a confirmation dialog. To avoid constant display of this dialog, use the Upload Information Management feature.

Using the Upload Information Management Feature

Before doing any modifications and at the initial start up of your system, do the following:

Step	Action
1	From the menu, select **Tools** →**Option** →**General tab**.
2	Select **Automatic** in the Upload Information Management area.
3	Press **OK** to close the window.
4	Save the program.
5	Download the program to the PLC.

Handling Online Modifications to the Standby CPU

For major modifications to the application program on the Standby CPU, verify that the Standby CPU is in Offline mode.

Two benefits result from this action:
- Run process continues
- Primary CPU does not perform a Switchover during modification of the Standby CPU

NOTE: If the Standby CPU is online during modifications, there is a possibility of Switchover occurring. If this occurs, the Standby CPU becomes Primary CPU and the process may run with incomplete modifications.

Performing Application Program Transfer

Avoid the possibility of having two different application programs running in the Primary CPU and Standby CPU by performing an application program transfer after completing online modifications with an application mismatch.

Resetting Command Register System Bit %SW60.3

When resetting the Command Register system bit `%SW60.3` to 0, you want to avoid the possibility of having two different application programs running in the Primary CPU and Standby CPU.

Step	Action
1	Connect to Primary CPU.
2	Access the Command Register system bit `%SW60.3`.
3	Reset bit to 0.

Firmware

6

Overview

This chapter describes the Quantum Hot Standby system firmware and how to upgrade it in the Standby CPU while the process is controlled by the Primary CPU.

What's in this Chapter?

This chapter contains the following topics:

Firmware Levels

Overview

The firmware level, selected in Unity Pro, defines the functionality of the Hot Standby CPU processor. There are major updates with new functions and minor releases with bug fixes.

If an application has a major firmware change in Unity Pro, the application must be completely rebuilt. Minor release changes do not require a rebuild.

Hot Standby CPU Firmware Levels

The following table gives the Quantum CPU firmware levels that allow construction of a Hot Standby system:

Firmware Version	Quantum CPUs	Function
2.00 to 2.60	140 CPU 671 60	Hot Standby
2.70	140 CPU 671 60	Hot Standby CCOTF *(see Quantum Ethernet I/O, System Planning Guide)*
2.80	140 CPU 671 60 140 CPU 672 61	Hot Standby CCOTF *(see Quantum Ethernet I/O, System Planning Guide)*
3.00	140 CPU 671 60 140 CPU 672 61	Hot Standby CCOTF *(see Quantum Ethernet I/O, System Planning Guide)* Quantum Ethernet I/O RIO

A given firmware level is backward compatibly, it has all the functions of the previous versions.

Hot Standby Coprocessor Firmware Levels

The following table gives the Quantum CPU Coprocessor firmware levels that are compatible with the CPU processor firmware:

CPU Firmware Version	Compatible Coprocessor Firmware Version	Recommended Coprocessor Version
2.11 to 2.42	2.11	2.11
2.50 to 2.51	2.50	2.50
2.60	2.60	2.60

CPU Firmware Version	Compatible Coprocessor Firmware Version	Recommended Coprocessor Version
2.70	2.70 to 2.79	2.71
2.80	2.80 to 2.89	2.80
3.00	3.00 to 3.09	3.00

Quantum Hot Standby Firmware Upgrade

Overview

The Firmware Upgrade feature allows the following upgrades while the Primary CPU controller continues to control the process:
- Operating System of the Standby CPU
- upgrading the firmware in the Standby coprocessor
- upgrading the firmware in the Standby CRP module

However, during the upgrade, the system is not redundant.

⚠ WARNING
UNEXPECTED EQUIPMENT BEHAVIOR
Design your application in such a way that your process is not impacted by a cycle time variation **that might appear after a firmware upgrade**.
Failure to follow these instructions can result in death, serious injury, or equipment damage.

Upgrading Firmware without Stopping

Under normal operating conditions, both controllers in a redundant system must have the same versions of firmware. There are checks by the controllers to detect if there is a mismatch in firmware.

Normally, when a mismatch exists, performing a Switchover is not be possible because the Standby CPU controller is not allowed to go to the RUN Primary mode.

However, to allow a firmware upgrade without stopping the application, set the Command Register system bit %SW60.4 *(see page 88)* to 1.

NOTE: Enabling the firmware upgrade without stopping the application overrides the process of checking whether the Primary CPU and Standby CPU are configured identically.

As soon as the firmware upgrade is finished set `%SW60.4` to 0 for an upgrade without stopping.

NOTE: An upgrade is only possible if the firmware used is compatible the target hardware.

Executing the Operating System Upgrade Procedure

General

Perform an Operating System upgrade using the OSLoader tool. Use one of two communication methods available in the OSLoader:
- Modbus RTU
- Modbus Plus

Using Modbus

List of the useful material:
- PC with Unity Pro and OSLoader
- cable 110 XCA 282 0• and adaptor 110 XCA 203 00

All the references about the keyboard are detailed in the Quantum Hardware Reference Manual:
- Controls and Displays *(see page 220)*
- Using the LCD display screens *(see page 224)*

To upgrade without stopping, refer to Upgrading the Operating System without Stopping *(see page 198)*.

Upgrade Procedure

When using Modbus or Modbus Plus, only address 1 is allowed for downloading. Ensure that no other device on the network uses address 1:

Step	Action
1	Connect Unity Pro to the Primary CPU (through Modbus, Modbus Plus or USB).
2	Set Command Register system bit `%SW60.4` to 1.
3	Disconnect Unity Pro from the Primary CPU.
4	Note the Modbus or Modbus Plus address of the Standby CPU using the keyboard functions. • for Modbus: **PLC Communications →Communications Serial Port** • for Modbus: **PLC Communications →Communications Modbus Plus**
5	Stop the Standby CPU with the keyboard functions. **NOTE:** The Standby CPU goes to STOP Offline mode; the Primary CPU operates as a Standalone CPU.
6	Disconnect all the communication links (Hot Standby fiber optic cable, Ethernet cables, Modbus Plus cables …) from the Standby rack.
7	Switch off the power of the Standby rack.
8	When using an application in the PCMCIA card: • Remove the PCMCIA card from the Standby CPU. • Remove the PCMCIA batteries to empty the card contents.
9	Power on the Standby CPU.

Step	Action
10	If not set to 1, change the Modbus or Modbus Plus address of the Standby CPU to 1 with the keyboard functions ● for Modbus: **PLC Communications →Communications Serial Port** ● for Modbus: **PLC Communications →Communications Modbus Plus**
11	Coprocessor Upgrade: **1.** Connect the PC to the Standby CPU using Ethernet (with appropriate switch and optical cable). **2.** Open the OSLoader tool. **3.** Select the FTP communication option. **4.** Connect the PC to the Standby CPU using the PLC IP address (read on the keypad). **5.** Download the Operating System to the Standby coprocessor. **6.** Power cycle the CPU.
12	CPU Operating System Upgrade: **1.** Connect the PC to the Standby CPU using Modbus or Modbus Plus. **2.** Open the OSLoader tool. **3.** Select the Modbus or Modbus Plus communication option. **4.** Connect to the Standby CPU using address 1. **5.** Download the Operating System to the Standby CPU.
13	Disconnect the PC from the Standby CPU.
14	Switch off the power of the Standby CPU.
15	When using an application in the PCMCIA: **1.** Insert the PCMCIA batteries. **2.** Insert the PCMCIA card in the Standby CPU.
16	Power on the Standby CPU. **NOTE:** The CPU must be in the `No Conf` state.
17	Check the Copro and Operating System versions in the CPU LCD Screen.
18	Reconnect all the communication cables (CRP module, Ethernet cables, …). Connect the Hot Standby Sync-link fiber optic cable last.
19	Check that the application program is automatically transferred to the Standby CPU. If not, perform the transfer with the keyboard. **NOTE:** Verify that the Modbus or Modbus Plus address is the same as the address indicated in Step 4.
20	Put the Primary and Standby CPUs in the **Run Primary** and **RUN Standby** Modes.
21	Perform a Switchover by stopping the Primary CPU with the keyboard. **NOTE:** Verify that the Standby CPU becomes Primary CPU (check the LCD screen).
22	Repeat Steps 4 through 21 on the new Standby CPU.
23	Connect Unity Pro to the new Primary CPU (through Modbus, Modbus Plus or USB).
24	Set Command Register system bit `%SW60.4` to 0.
25	Disconnect the PC and ensure Primary and Standby CPUs are in **Run Primary** and Run Standby Modes.

Compatibility Issues

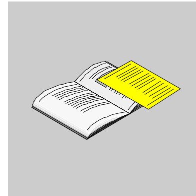
To upgrade a Quantum Hot Standby Operating System without shutting down the process, the current application program must be executable by the new Operating System.

Observe this requirement when installing minor revisions targeted for bug fixes or minor enhancements.

When a major function enhancement needs to be made, maintaining this compatibility may not be possible.

In this case, to perform an Operating System upgrade requires a system shut down.

Appendices



At a Glance

The appendices for the Quantum Hot Standby system are included here.

What's in this Appendix?

The appendix contains the following chapters:

Quantum Hot Standby Additional Information

A

Overview

This appendix describes the necessary cables, design specifications, error codes.

What's in this Chapter?

This chapter contains the following topics:

Fiber Optic Sync-Link Cable in a Hot Standby System

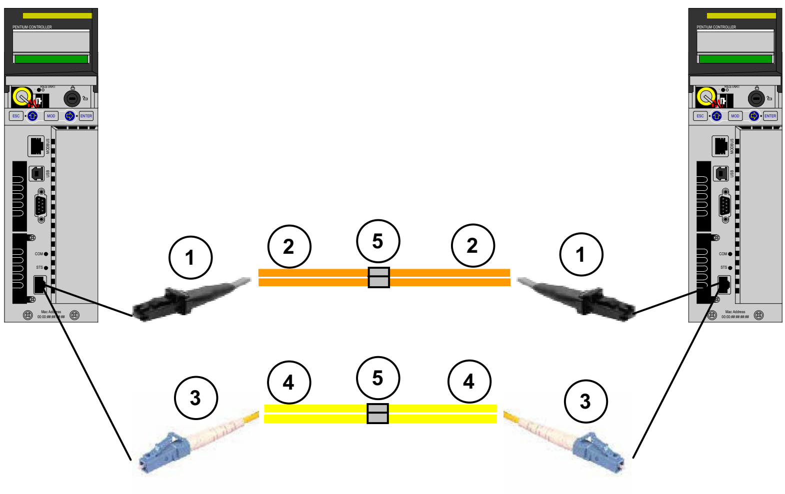
Schneider Electric Recommends

Recommendations:

- For 140 CPU 671 60 modules, use up to 4 km (2.5 mi) of 62.5/125 µm, graded index, duplex, multi mode glass fiber (usually referred to as OM1 type fiber).
  This type of fiber is rated at maximum attenuation of 1.5 dB per km (maximum, at 1300 nm).
- For 140 CPU 672 61 modules, use up to 16 km (9.9 mi) of 9/125 µm, duplex, single mode glass fiber (usually referred to as OS1 or G652 type fiber).
  This type of fiber is rated at maximum attenuation of 0.35 dB per km (maximum, at 1300 nm).
- Wherever possible, use a multifiber cable since the cable is less expensive and provides a backup in case one of the fibers is cut during installation.

Typical Configuration Scheme

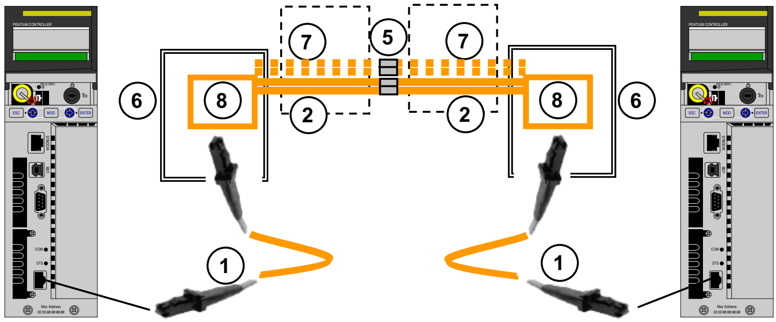
The following figure represents the direct connection with splices between two CPUs:



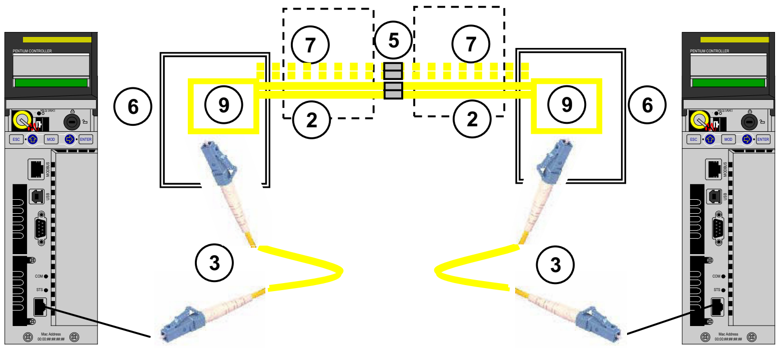
Explanation of direct connections above:

1. MTRJ connector
2. Duplex 62.5/125 µm, graded index, multi-mode fiber optic cable
   Use only single mode with 140 CPU 671 60: up to 4 km (2.5 mi).
3. LC connector
4. Duplex 9 / 125 µm, single-mode fiber optic cable
   Use only single mode with 140 CPU 672 61: up to 16 km (9.9).
5. Splices

The following figures represent the direct connection with splices between two modules when using a multi-fiber cable:



Multimode (140 CPU 671 60 CPUs): up to 4 km (2.5 mi)



Single mode (140 CPU 672 61 CPUs): up to 16 km (9.9 mi)

Explanation of direct connections above:

1. MTRJ/MTRJ fiber connector
2. Duplex 62.5/125 µm, graded index, multi-mode fiber optic Cable
3. LC/LC fiber connector
4. Duplex 9/125 µm, single-mode fiber optic Cable
5. Splices
6. Fiber distribution box
7. Backup fiber
8. MTRJ jack (or MTRJ coupler)
9. LC jack (or LC/LC coupler)

Optical Power Budget Calculation

The maximum length of Hot Standby fiber optic link must be calculated by considering total loss in all components used in the path, fiber optic cable, optical connectors and splices:

- For 140 CPU 671 60 PLCs, the Power Loss Budget in 62.5/125 µm fiber cable equals 9.9 dB (including system margin).
- For 140 CPU 672 61 PLCs, the Power loss Budget in 9/125 µm fiber cable equals 9 dB (including system margin)

$$\text{Max distance} = \frac{\text{Power Loss Budget [dB] - number of connectors} \times \text{0.35dB - number of splices} \times \text{0.15dB)}}{\text{fiber attenuation [dB/km]}}$$

NOTE: There is no minimum distance requirement.

Cables Available

The following cables are available from Schneider Electric:

Multi-mode part numbers for 140 CPU 671 60	Description
490 NOR 000 03	3 m MTRJ/MTRJ
490 NOR 000 05	5 m MTRJ/MTRJ
490 NOR 000 15	15 m MTRJ/MTRJ

Single-mode part numbers for 140 CPU 672 61	Description
VDIF0646463505	5 m LC/LC

140 CPU 671 60 Specifications

Module Specifications

Element	Description
Communication ports	1 Modbus (RS-232/RS-485) 1 Modbus Plus (RS-485) 1 USB 1 Ethernet (used as a Hot Standby port)
Bus current required	2.5 A
Maximum number of NOM, NOE, PTQ PDP MV1 and MMS modules supported (any combination)	6
Key switch	Yes
Keypad	Yes

Processor

Function	Description
Model	Punting
Clock speed	266 MHz
Coprocessor	Yes, Built-in Ethernet
Watchdog timer	250 ms software adjustable

Memory

RAM	2 MByte
IEC program memory (and/or application data and configuration IEC program memory (maximum with PCMCIA card)	1024 kByte 7168 kBytes

Program Execution Time

Kilo Instruction executed per millisecond (Kins/ms)		Execution time per instruction (ms/Kins)	
100% Boolean	65% Boolean + 35% digital	100% Boolean	65% Boolean + 35% digital
10.28	10.07	0.097	0.099

NOTE: When considering the execution time with the RAM or the PCMCIA card, the values are identical as when the program execution takes place within the CACHE memory.

Reference Capacity

Discrete (bits)	64 k (any combination)
Registers (words)	64 k max.

Remote I/O

Max. I/O words/drop	64 in / 64 out*
Max. number of remote drops	31
* This information can be a mix of discrete or register I/O. For each word of configured I/O, one of the I/O words must be subtracted from the total available.	

Battery and Clock

Battery type	3 V Lithium
Service life	1.2 Ah
Shelf life	10 years with 0.5% loss of capacity/year
Battery load current @ power-off	typical: 14 µA
	maximum 420 µA
TOD clock	+/-8.0 s/day @ 0 .. 60 °C

Diagnostic

Power-up	RAM RAM address Executive Checksum User Logic Check Processor
Run Time	RAM RAM address Executive Checksum User Logic Check

140 CPU 671 60S Specifications

Module Specifications

Component	Description
Communication ports	1 Modbus (RS-232/RS-485) 1 Modbus Plus (RS-485) 1 USB 1 Ethernet (used as a Hot Standby port)
Bus current required	2.5 A
Maximum number of NOE 771 11 modules supported	6
Key switch	Yes
Keypad	Yes

Processor

Feature	Description
Model	Pentium
Clock speed	266 MHz
Coprocessor	Yes, Built-in Ethernet
Watchdog timer	250 ms software adjustable

Memory

RAM	4 MByte
IEC program memory (and/or application data and configuration) IEC program memory (maximum with PCMCIA card)	1024 kByte 7168 kBytes

Reference Capacity

Discrete (bits)	64 k (any combination)
Registers (words)	64 k maximum

Remote I/O

Maximum I/O words/drop	64 in / 64 out*
MAXIMUM number of remote drops	31
* This information can be a mix of discrete or register I/O. For each word of configured I/O, one of the I/O words must be subtracted from the total available.	

Battery and Clock

Battery type	3 V Lithium
Service life	1.2 Ah
Shelf life	10 years with 0.5% loss of capacity/year
Battery load current @ power-off	typical: 14 µA
	maximum 420 µA
TOD clock	+/-8.0 s/day @ 0 .. 60 °C

Diagnostic

Power-up	RAM RAM address Executive Checksum User Logic Check Processor
Run Time	RAM RAM address Executive Checksum User Logic Check

140 CPU 672 61 Specifications

Module Specifications

Component	Description
Communication ports	1 Modbus (RS-232/RS-485) 1 Modbus Plus (RS-485) 1 USB 1 Ethernet (used as a Hot Standby port)
Bus current required	2.5 A
Maximum number of NOE 771 11 modules supported	6
Key switch	Yes
Keypad	Yes

Processor

Feature	Description
Model	Pentium
Clock speed	266 MHz
Coprocessor	Yes, Built-in Ethernet
Watchdog timer	250 ms software adjustable

Memory

RAM	4 MBytes
IEC program memory (and/or application data and configuration) IEC program memory (maximum with PCMCIA card)	3172 kBytes 8 MBytes

Reference Capacity

Discrete (bits)	64 kBytes (any combination)
Registers (words)	64 kBytes maximum

Remote I/O

Maximum I/O words/drop	64 in / 64 out*
Maximum number of remote drops	31
* This information can be a mix of discrete or register I/O. For each word of configured I/O, one of the I/O words must be subtracted from the total available.	

Battery and Clock

Battery type	3 V Lithium
Service life	1.2 Ah
Shelf life	10 years with 0.5% loss of capacity/year
Battery load current @ power-off	typical: 14 µA
	maximum 420 µA
TOD clock	+/-8.0 s/day @ 0 .. 60 °C

Diagnostic

Power-up	RAM address Executive Checksum User Logic Check Processor
Run Time	RAM address Executive Checksum

CRP Remote I/O Head Processor Detected Error Patterns

Detected Error Patterns

The following table displays both:
- Number of times the **Com Act** indicator blinks for each type of error
- Possible codes for each type of blink

The detected errors:

Number of blinks on Com Act Indicator	Code in hex	Detected Error
Slow (steady)	0000	Requested kernel mode
2	6820	Detected HCB frame pattern error
	6822	Detected head control block diag error
	6823	Detected mod personality diag error
	682A	Detected fatal start IO error
	682B	Incorrect read I/O pers request
	682C	Incorrect execute diag request
	6840	ASCII input transfer state
	6841	ASCII output transfer state
	6842	IO input comm state
	6843	IO output comm state
	6844	ASCII abort comm state
	6845	ASCII pause comm state
	6846	ASCII input comm state
	6847	ASCII output comm state
	6849	Building 10 byte packet
	684A	Building 12 byte packet
	684B	Building 16 byte packet
	684C	Illegal I/O drop number
3	6729	984 interface bus ack stuck high
4	6616	Detected coax cable initialization error
	6617	Detected coax cable DNA transfer error
	6619	Detected coax cable dumped data error
	681A	Coax cable DRQ line hung
	681C	Coax cable DRQ hung
5	6503	Detected RAM address test error
6	6402	Detected RAM data test error

Number of blinks on Com Act Indicator	Code in hex	Detected Error
7	6300	Detected PROM checksum error (OS not loaded)
	6301	Detected PROM checksum error
8	8001	Detected kernel PROM checksum error
	8002	Detected flash prog/erase error
	8003	Unexpected OS return

TextIDs

TextIDs Definitions

TextIds define the warning messages written in the diagnostic buffer.

TextIDs for switching from Primary CPU to Offline:

TextID	Warning message
13001	System halt
13002	Remote IO error
13003	ETH device error
13004	ETH communication problem
13005	Stop PLC command
13006	Offline keypad switch
13007	Offline Command register request

TextIDs for switching from Standby CPU to Offline:

TextID	Warning message
13008	System halt
13009	Remote IO error
13010	ETH device error
13011	ETH communication problem
13012	Stop PLC command
13013	Offline keypad switch
13014	Offline Command register request

TextIDs for switching from Standby CPU to Primary CPU:

TextID	Warning message
13015	Control command over ETH
13016	Control command over RIO

TextIDs for switching from Offline to Primary CPU/Standby CPU:

TextID	Warning message
13017	Switch from Offline to Primary CPU
13018	Switch from Offline to Standby CPU

Quantum Hot Standby Controls, Displays and Menus

B

Overview

This appendix describes controls and displays, LED descriptions and the structure of the screen menus.

What's in this Chapter?

This chapter contains the following topics:

CPU Controls and Displays

Lens Cover

The protective lens cover (2 in the CPU front panel *(see page 26)*) can be opened by sliding upwards.

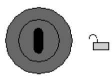
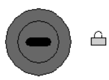
With the lens cover open you have access to the following items:
- key switch
- battery
- reset button

Key Switch

The key switch (4) is a security feature and a memory protection switch. The key switch has two positions: locked and unlocked. The key switch is only read and deciphered by the PLC OS portion of the firmware and not by the OS loader portion.

The Quantum processors have a set of system menus that enable the operator to:
- perform PLC operations (i.e., start PLC, stop PLC)
- display module parameters (i.e., communications parameters)
- switch to the maintenance mode (in Safety processors)

The key positions are shown in the table below:

Key Position	PLC Operation
unlocked: 	• System menu operations can be invoked and module parameters can be modified by the operator with the LCD and keypad. • Memory protection is OFF. • You can switch to Maintenance mode (in Safety processors).
locked: 	• No system menu operations can be invoked and module parameters are read-only. • Memory protection is ON. • Safe mode forced (in Safety processors).

Switching the key switch position from locked to unlocked or vice versa turns on the LCD's backlight.

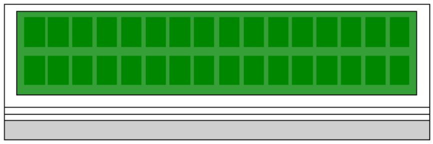
NOTE: For more explanations about Maintenance and Safe mode of Safety processors *(see Modicon Quantum, Quantum Safety PLC, Safety Reference Manual)*

Reset Button

Pressing the reset button (12) forces a cold start of the PLC.

LCD Display

The liquid crystal display (LCD - 3) has 2 lines with 16 characters each with changeable backlight state and contrast:



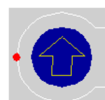
The backlight handling is entirely automated to save the life of the LCDs. The backlight turns on when one of the following occurs:

- a key is pressed
- the key switch state is changed
- an error message is displayed on the LCD

The backlight stays on for error messages as long as the error message is displayed otherwise, the backlight automatically turns off after five minutes.

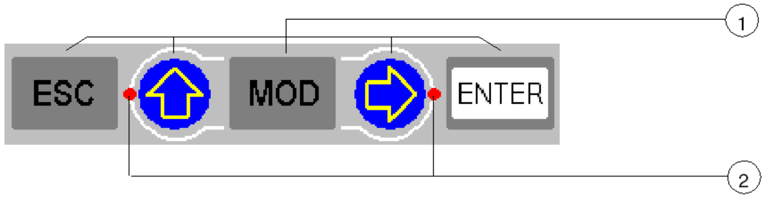
Adjusting the Contrast

The contrast is adjustable from the keypad when the default screen is displayed:

Step	Action	
1	Press the `MOD` key:	
2	To adjust the contrast darker press:	
3	To adjust the contrast lighter press:	
4	To confirm the setting press:	

Keypad

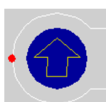
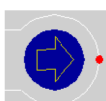
The keypad (5) has five keys that are mapped to hardware addresses. Each of the two arrow keys includes an LED:



1 5 keys
2 2 LEDs

Using the Keys

Keypad functions:

Key	Function	
	To cancel an entry, suspend or stop an action in progress To display the preceding screens successively (move up the menu tree)	
	To confirm a selection or an entry	
	To set a field on the display into the modify mode	
	LED: on	Key active: ● to scroll through menu options ● to scroll through modify mode field options
	LED: flashing	Key active: The modify mode field can be scrolled.
	LED: off	Key inactive: No menu options and no field options.
	LED on	Key active: ● to move around in a screen, field to field ● to go to the sub-menu
	LED flashing	Key active: Used to move digit to digit in a modify mode field.
	LED off	Key inactive, there is no: ● sub-menu for menu option ● scrolling around a screen ● scrolling around a field

CPU LED Indicators

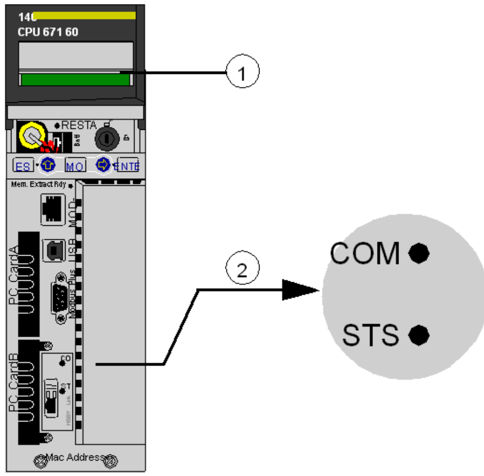
Overview

The Quantum Hot Standby CPUs has two types of indicators:

1. LCD display screen (1)
The default display screen serves as a controller status screen *(see page 224)*.

2. LED indicators (2)

The following figure shows the two types of indicators:



1 LCD display (lens cover closed)
2 LED indicators

LED Description

The following table shows the description for the Quantum Hot Standby CPU LED indicators:

LEDs	Indication
COM	CPU is controlled by the Copro hardware, indicates Primary or Standby CPU activity
STS	CPU is controlled by the Copro firmware: ● Blinking: system is redundant and data are exchanged between the Primary and Standby PLCs ● ON: system not redundant / Copro booting from power-on to end of self-tests ● OFF: Copro auto-test detected errors

Using the CPU LCD Display Screens

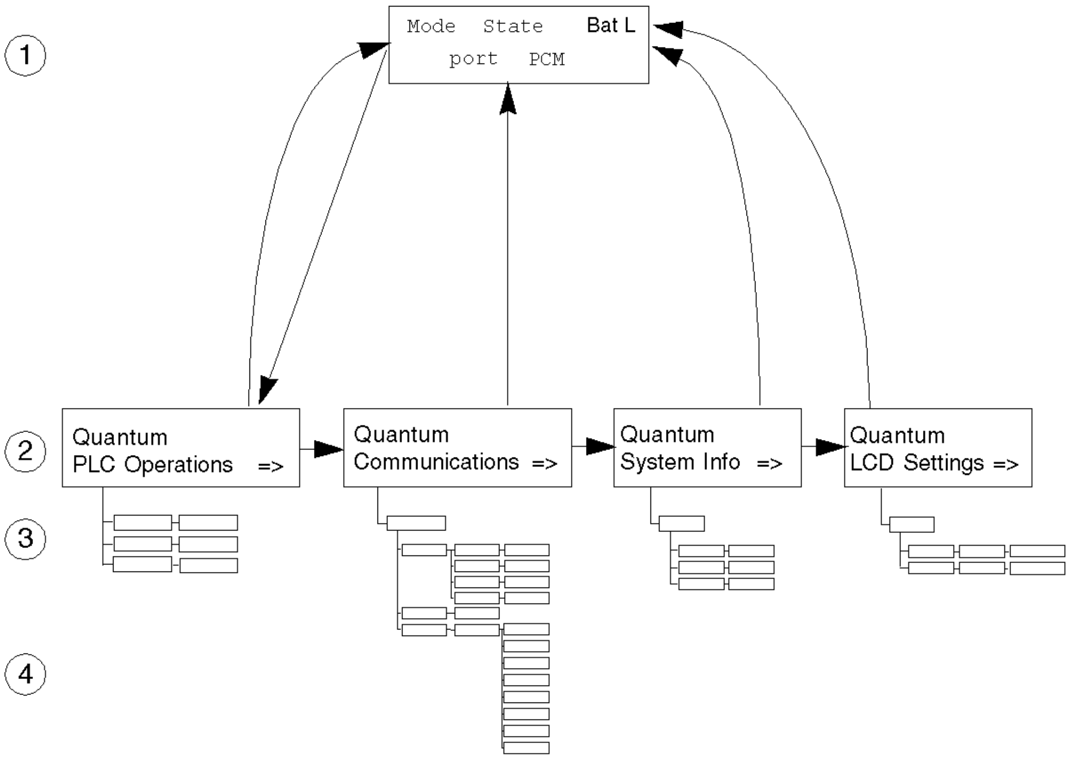
Overview

The controller LCD displays messages. These messages indicate the controller status. There are four levels of menus and submenus. Menus are accessed using the keypad *(see page 222)* on the front of the controller.

For detailed information about the menus and submenus see:

- PLC Operations Menus and Submenus *(see page 227)*
- Using the Communications Menus and Submenus *(see page 230)*
- Using the LCD Settings Menus and Submenus *(see page 232)*
- Using the System Info Menus and Submenus *(see page 233)*

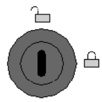
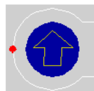
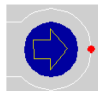
Structure: LCD display menus and submenus:



1 Default screen
2 System menus
3 Sub menus
4 Sub screens

Accessing the Screens

Use the keys on the keypad to access the system menus and submenus:

Step	Action
1	To access the screens, ensure that the key switch is in the unlocked position. 
2	To step down to a lower menu, operate one of the following keys:   
3	To return to the previous menu, press: 

Default Screen

The default screen is read-only and contains the following fields:

```
Mode   State    Bat L
   port    PCM
```

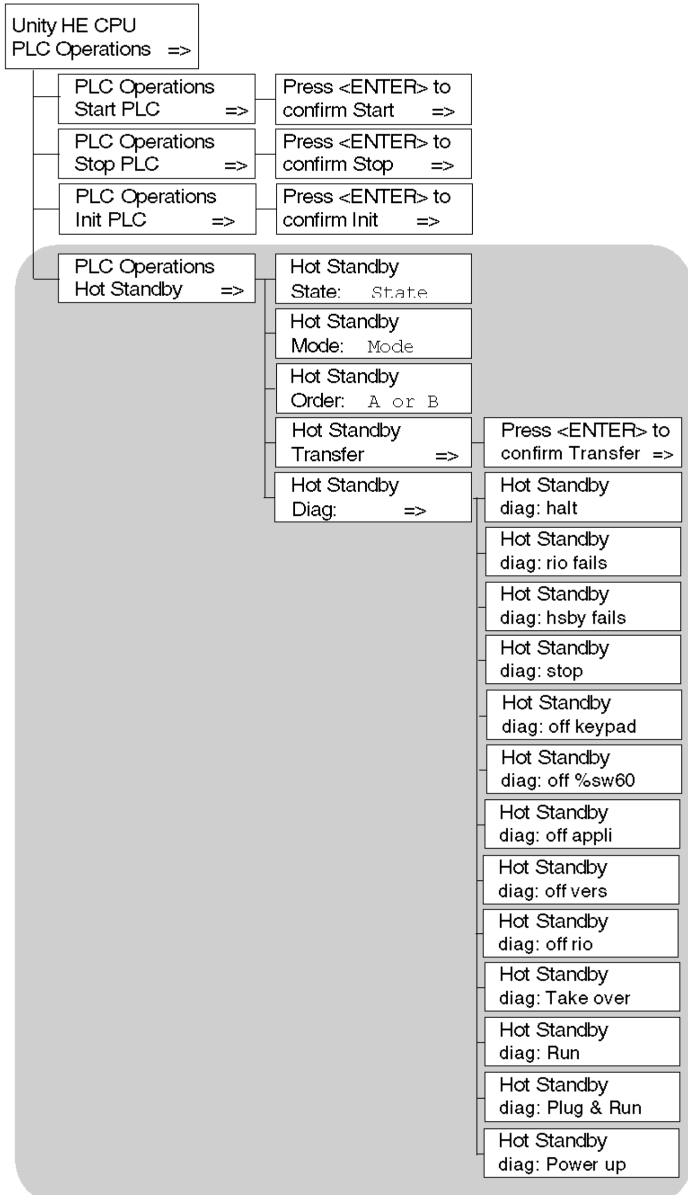
The default screen displays the following information:

Fields Available	Options Available	Description
`Mode`	M	Maintenance Mode (only on safety processors)
	S	Safe Mode (only on safety processors)
`State`	RUN	Application program is running.
	RUN Prim	RUN as Primary CPU processor (only on Hot Standby processors)
	RUN Stby	RUN as Standby CPU processor (only on Hot Standby processors)
	RUN OffL	RUN Offline (Hot Standby processor not connected to another processor)
	STOP	Application program is NOT running
		STOP Offline
	No Conf	Processor has no application program
	Halt	Detected state error (in maintenance mode for safety modules)

Fields Available		Options Available	Description
BatL			Indicates battery health: ● Steady = battery is low ● No message = battery is OK
Port	USB		Indicates that the port has activity
	Modbus Plus	MB+	Indicates Modbus Plus activity
		mb+	No activity
		Dup	Duplicate MB+ address
		ERR	Detected Modbus communications error
		INI	Initial Network Search
	Modbus	232	Serial port activity for RS-232
		485	Serial port activity for RS-485
	PCM	1	Displayed status indicates battery health of the PCMCIA card in slot 1: ● Steady = battery is OK ● Flashing = battery is low (only for green PCMCIAs (version <04)) *
		2	Displayed status indicates battery health of the PCMCIA card in slot 2: ● Steady = battery is OK ● Flashing = battery is low (only for green PCMCIAs (PV < 04)) *
		* With blue PCMCIAs (version >= 04), when main battery is low there is no flash.	

PLC Operations Menu

The structure of the PLC Operations menu and submenus are:



Submenu for PLC Operations: Start, Stop and Init:

Start, Stop, Init Screens Display	Fields Available	Description
Start PLC	Press <ENTER> to confirm Start	Pressing <ENTER> starts the controller
Stop PLC	Press <ENTER> to confirm Stop	Pressing <ENTER> stops the controller
Init PLC	Press <ENTER> to confirm Init	Pressing <ENTER> initializes the controller On safety processors, this command is only available in the maintenance mode.

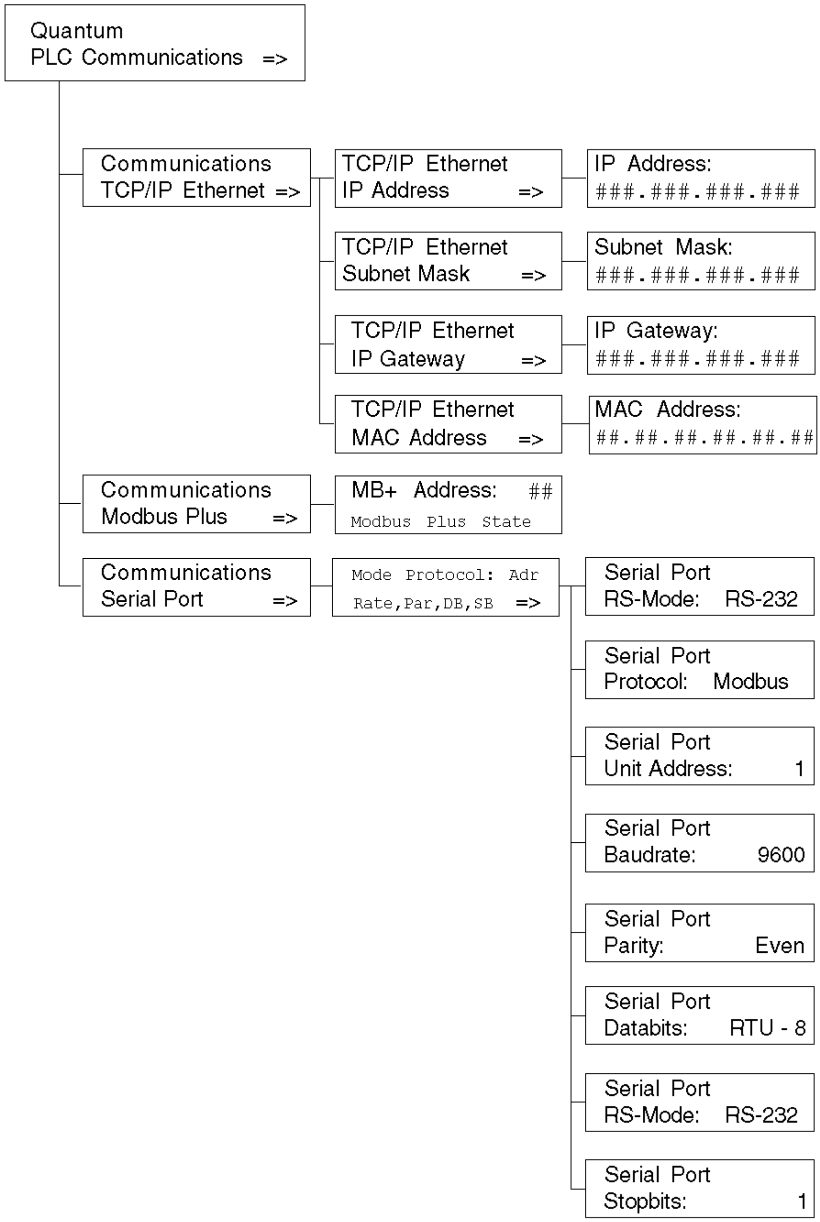
Submenu for PLC Operations Hot Standby CPU:

<table>
<tr><th>Screen</th><th>Field</th><th colspan="2">Option</th><th>Description</th></tr>
<tr><td rowspan="3">Hot Standby State:</td><td rowspan="3"><code>State</code>
read only</td><td colspan="2">PRIMARY CPU</td><td>Controller serves as Primary unit</td></tr>
<tr><td colspan="2">STANDBY CPU</td><td>Controller serves as Standby unit</td></tr>
<tr><td colspan="2">Offline</td><td>Controller is neither Primary nor Standby unit</td></tr>
<tr><td rowspan="4">Hot Standby Mode:</td><td rowspan="4"><code>Mode</code>
(modifiable only if the key switch is in the unlocked position)</td><td rowspan="2">RUN</td><td>STS steady</td><td>Controller is active and is either serving as Primary PLC or able to take over the Primary CPU role if needed</td></tr>
<tr><td>STS flashing</td><td>Controller is transferring/updating. When the transfer is done, RUN stays on steady</td></tr>
<tr><td rowspan="2">OFFLINE</td><td>STS steady</td><td>Controller is taken out of service without stopping it or disconnecting it from power.
If the controller is the Primary PLC when the mode is changed to Offline, control switches to the Standby PLC.
If the Standby PLC changes to Offline, the Primary CPU continues to operate without a backup.</td></tr>
<tr><td>STS flashing</td><td>Controller is transferring/updating.When the transfer is done, OFFLINE stays on steady.</td></tr>
<tr><td rowspan="2">Hot Standby Order:</td><td rowspan="2"><code>A or B</code>
(Modifiable only if the key switch is in the unlocked position.)</td><td colspan="2">FIRST</td><td rowspan="2">Hot Standby Power Order
NOTE: To change the A/B order the PLC must be in the STOP mode.</td></tr>
<tr><td colspan="2">SECOND</td></tr>
<tr><td>Hot Standby Transfer:</td><td colspan="3">-
(This menu option is only enabled, if the key switch is in the unlocked position)</td><td>Pressing the <ENTER> key confirms the Transfer. The transfer initiates the request of a program update from the Primary PLC.
Pressing any other key cancels the Transfer initiation and returns to the Hot Standby Transfer menu option screen.</td></tr>
</table>

Screen	Field	Option	Description
Hot Standby Diag:	The order of diagnostic screen varies with the operation.		
	Halt		User task in halt mode
	RIO fails		Detected error reported by RIO head
	HSBY fails		Detected error reported by optical link
	Stop		Stop command sent
	Off keypad		Offline command entered on keypad
	Off %SW60		Offline command set in command register
	Off appli		Offline due to application mismatch
	Off vers		Offline due to PLC or Copro OS mismatch
	Off RIO		Offline due to Remote I/O error
	Take over		Standby CPU switched to Primary CPU mode
	Run		Run command sent
	Plug & Run		Sun-link operational and Standby CPU is started
	Power up		No message: PLC has just started

Communications Menu

Communications menu and submenus:



Submenu for TCP/IP Ethernet PLC Communications submenus:

TCP/IP Ethernet Screen Displays	Fields Available	Options Available	Description
TCP/IP Ethernet IP Address[1,2]	###.###.###.###	decimal numbers	Displays IP address
TCP/IP Ethernet Subnet Mask[1,2]	###.###.###.###	decimal numbers	Displays Subnetwork Mask address
TCP/IP Ethernet IP Gateway[1,2]	###.###.###.###	decimal numbers	Displays Ethernet IP Gateway address
TCP/IP Ethernet MAC Address	##.##.##.##.##.## (read only)	hexadecimal numbers	Displays MAC (Medium Access Control) address

1) Parameters can be modified only if no applications have been downloaded (in NO CONF state).

2) When a new PLC application has been downloaded, the Ethernet address on the screen is only updated after accessing the highest level of the menu structure.

Modbus Plus PLC Communications submenus:

Fields Available	Options Available	Description
## (Modifiable only if the key switch is in the unlocked position.)	1-64	Enter a valid Modbus Plus address
Modbus Plus State	Monitor Link	Modbus Plus State
	Normal Link	
	Sole Station	
	Duplicate address	
	No Token	

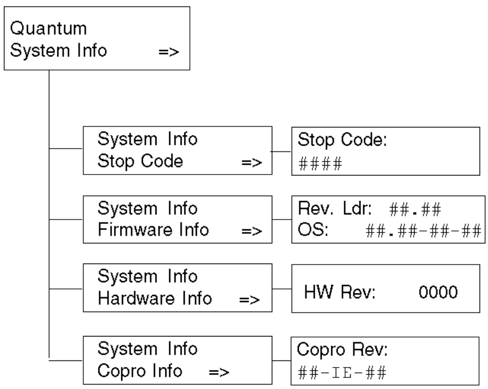
Serial PLC Communications submenus:

Fields Available*	Options Available	Description
Mode	232	RS mode
	485	
Protocol	ASCII	Protocols available
	RTU	
Adr	1 - 247	Unit address
	for Modbus Switchover Primary CPU 1-119 Standby CPU 129 - 247	

Fields Available*	Options Available	Description
Rate	50, 75, 110, 134.5, 150, 300, 600, 1200, 1800, 2400, 3600. 4800, 7200, 9600, 19200 bits/s	Baud rate
Par	NONE	Parity
	ODD	
	EVEN	
DB	7,8	Data bits, if Protocol is Modbus, then RTU-8 or ASCII-7.
SB	1,2	Stop bits
*If the key switch is in the unlocked position, fields are modifiable.		

System Info Menu

Structure of System Info menus and submenus:

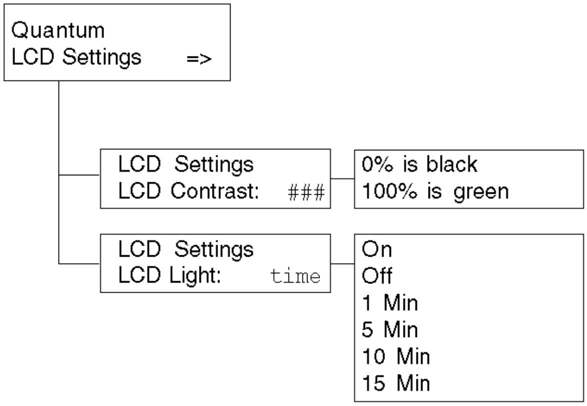


System Info, PLC Communications submenus:

System Info Screen Displays	Fields Available*	Option Available	Description
Stop Code	####		Machine stop code
	Description		Description of the machine stop code
Firmware Info	Rev.Ldr: ##.##		OS revision
	OS: ##.##-##-##		OSLoader revision
Hardware Info	HW Rev: ####		Hardware revision
Copro Info	##-IE-##		Copro revision
*Fields are read only.			

LCD Settings Menu

LCD Settings menus and submenus:



LCD Contrast settings submenu:

LCD Screen Contrast Screen Displays	Fields Available	Description
LCD Contrast:	`####`	Use the arrow keys to adjust the setting: ● Up arrow increases percent (brighter) ● Right arrow decreases percent (darker)

LCD Light setting submenus:

Screen Displays	Fields Available	Description
LCD Light:	`On`	LCD remains on permanently or until changed.
	`Off`	LCD remains off permanently or until changed.
	`1 Min`	LCD remains on for 1 minute.
	`5 Min`	LCD remains on for 5 minutes.
	`10 Min`	LCD remains on for 10 minutes.
	`15 Min`	LCD remains on for 15 minutes.

Glossary

0-9

%I

According to the CEI standard, `%I` indicates a language object of type discrete IN.

%IW

According to the CEI standard, `%IW` indicates a language object of type analog IN.

%M

According to the CEI standard, `%M` indicates a language object of type memory bit.

%MW

According to the CEI standard, `%MW` indicates a language object of type memory word.

%Q

According to the CEI standard, `%Q` indicates a language object of type discrete OUT.

%QW

According to the CEI standard, `%QW` indicates a language object of type analog OUT.

%SW

According to the CEI standard, `%SW` indicates a language object of type system word.

A

adapter

The target of real-time I/O data connection requests from scanners. It cannot send or receive real-time I/O data unless it is configured to do so by a scanner, and it does not store or originate the data communications parameters necessary to establish the connection. An adapter accepts explicit message requests (connected and unconnected) from other devices.

advanced mode

A selection in Unity Pro that displays expert-level configuration properties that help define Ethernet connections. Because these properties should be edited only by people with a good understanding of EtherNet/IP communication protocols, they can be hidden or displayed, depending upon the qualifications of the specific user.

architecture

A framework for the specification of a network, constructed on the following:
- physical components and their functional organization and configuration
- operational principles and procedures
- data formats used in its operation

array

A table containing elements of a single type.

The syntax is as follows: `array [<limits>] OF <Type>`

Example:

`array [1..2] OF BOOL` is a one-dimensional table with two elements of type `BOOL`.

`array [1..10, 1..20] OF INT` is a two-dimensional table with 10x20 elements of type `INT`.

ART

(*application response time*) The time a PLC application takes to react to a given input. ART is measured from the time a physical signal in the PLC turns on and triggers a write command until the remote output turns on to signify that the data has been received.

B

BOOL

(*boolean type*) The basic data type in computing. A `BOOL` variable can have either of the following two values: 0 (`FALSE`) or 1 (`TRUE`).

A bit extracted from a word is of type `BOOL`, for example: `%MW10.4`.

BOOTP

(*bootstrap protocol*) A UDP network protocol that can be used by a network client to automatically obtain an IP address from a server. The client identifies itself to the server using its MAC address. The server, which maintains a pre-configured table of client device MAC addresses and associated IP addresses, sends the client its defined IP address. The BOOTP service utilizes UDP ports 67 and 68.

broadcast

A message sent to all devices in the subnet.

C

CCOTF

(*change configuration on the fly*) A feature of Unity Pro that allows a PLC hardware change in the system configuration while the PLC is operating and not impacting other active drop operations.

CIP™

(*common industrial protocol*) A comprehensive suite of messages and services for the collection of manufacturing automation applications — control, safety, synchronization, motion, configuration and information. CIP allows users to integrate these manufacturing applications with enterprise-level Ethernet networks and the internet. CIP is the core protocol of EtherNet/IP.

class 1 connection

A CIP transport connection used for I/O data transmission via implicit messaging between EtherNet/IP devices.

class 3 connection

A CIP transport connection used for explicit messaging between EtherNet/IP devices.

connected messaging

Using a CIP connection for communication that establishes a relationship between 2 or more application objects on different nodes. The connection establishes a virtual circuit in advance for a particular purpose, such as frequent explicit messages or real-time I/O data transfers.

connection

A virtual circuit between 2 or more network devices, created prior to the transmission of data. After a connection is established, a series of data is transmitted over the same communication path, without the need to include routing information — including source and destination address — with each piece of data.

connection originator

The EtherNet/IP network node that initiates a connection request for I/O data transfer or explicit messaging.

connectionless

Communication between 2 network devices, where data is sent without prior arrangement between the devices. Each piece of transmitted data includes routing information — including source and destination address.

control network

An Ethernet-based network containing PLCs, SCADA systems, an NTP server, PCs, AMS, switches, etc. Two kinds of topologies are supported:
- flat — Devices in this network belong to the same subnet.
- 2 levels — The network is split into an operation network and an inter-controller network. These 2 networks can be physically independent, but are generally linked by a routing device.

D

DDT

(*derived data type*) A set of elements with the same type (`array`) or with different types (structure).

determinism

For a defined application and architecture, the ability to predict that the delay between an event (change of an input value) and the corresponding change of an output state is a finite time *t*, smaller than the time required for your process to run correctly.

device network

An Ethernet-based network within a remote I/O network that contains both remote I/O and distributed I/O devices. Devices connected on this network follow specific rules to allow remote I/O determinism.

DFB

(*derived function block*) Function blocks that can be defined by the user in ST, IL, LD or FBD language.

Using these DFB types in an application makes it possible to:
- simplify the design and entry of the program
- make the program easier to read
- make it easier to debug
- reduce the amount of code generated

DHCP

(*dynamic host configuration protocol*) An extension of the BOOTP communications protocol that provides for the automatic assignment of IP addressing settings—including IP address, subnet mask, gateway IP address, and DNS server names. DHCP does not require the maintenance of a table identifying each network device. The client identifies itself to the DHCP server using either its MAC address, or a uniquely assigned device identifier. The DHCP service utilizes UDP ports 67 and 68.

distributed I/O cloud

A group of distributed I/O devices connected either to a non-ring port on a DRS or to a distributed I/O communications module in the local rack. Distributed I/O clouds are single-point connections to the Quantum EIO network and are not required to support RSTP.

distributed I/O device

Any Ethernet device (Schneider Electric device, PC, servers, or third-party devices) that supports I/O exchange with a PLC or other Ethernet communication service.

distributed I/O network

A network containing distributed I/O devices that integrates a unique standalone PLC or a unique Hot Standby system. I/O scanning may be performed by a communication module interlinked with a remote I/O head module on the local rack of an Ethernet remote I/O system. Distributed I/O network traffic is delivered after remote I/O traffic, which takes priority in an Ethernet remote I/O network.

DNS

(*domain name server/service*) A service that translates an alpha-numeric domain name into an IP address, the unique identifier of a device on the network.

domain name

An alpha-numeric string that identifies a device on the internet, and which appears as the primary component of a web site's uniform resource locator (URL). For example, the domain name *schneider-electric.com* is the primary component of the URL *www.schneider-electric.com*.

Each domain name is assigned as part of the domain name system, and is associated with an IP address.

Also called a host name.

DRS

(*dual-ring switch*) A ConneXium extended managed switch with one of several possible predefined configurations downloaded to it so that it can participate in a Quantum EIO network. A DRS provides 2 RSTP-enabled ring connections, one for the main ring and one for a sub-ring. It also manages QoS, which provides a predictable level of performance for both remote I/O and distributed I/O traffic on the same I/O network.

DRSs require a firmware version 6.0 or later.

DT

(*date and time*) A data type encoded in BCD in a 64-bit format that contains the following information:
- the year encoded in a 16-bit field
- the month encoded in an 8-bit field
- the day encoded in an 8-bit field
- the time encoded in an 8-bit field
- the minutes encoded in an 8-bit field
- the seconds encoded in an 8-bit field

NOTE: The 8 least significant bits are not used.

The `DT` type is entered as follows:

`DT#<Year>-<Month>-<Day>-<Hour>:<Minutes>:<Seconds>`

This table shows the upper/lower limits of each field:

Field	Limits	Comment
Year	[1990,2099]	Year
Month	[01,12]	The leading 0 is displayed; it can be omitted during data entry.
Day	[01,31]	For months 01/03/05/07/08/10/12
	[01,30]	For months 04/06/09/11
	[01,29]	For month 02 (leap years)
	[01,28]	For month 02 (non-leap years)
Hour	[00,23]	The leading 0 is displayed; it can be omitted during data entry.
Minute	[00,59]	The leading 0 is displayed; it can be omitted during data entry.
Second	[00,59]	The leading 0 is displayed; it can be omitted during data entry.

DTM

(*device type manager*) A device driver running on the host PC. It provides a unified structure for accessing device parameters, configuring and operating the devices, and troubleshooting the network. DTMs can range from a simple graphical user interface (GUI) for setting device parameters to a highly sophisticated application capable of performing complex real-time calculations for diagnosis and maintenance purposes. In the context of a DTM, a device can be a communications module or a remote device on the network.

See *FDT*.

E

EDS

(*electronic data sheet*) Simple text files that describe the configuration capabilities of a device. EDS files are generated and maintained by the manufacturer of the device.

EF

(*elementary function*) A block used in a program to perform a predefined logical function.

A function does not have any information on the internal state. Several calls to the same function using the same input parameters will return the same output values. You will find information on the graphic form of the function call in the [*functional block (instance)*]. Unlike a call to a function block, function calls include only an output which is not named and whose name is identical to that of the function. In FBD, each call is indicated by a unique [number] via the graphic block. This number is managed automatically and cannot be modified.

Position and configure these functions in your program in order to execute your application.

You can also develop other functions using the SDKC development kit.

EFB

(*elementary function block*) A block used in a program to perform a predefined logical function.

EFBs have states and internal parameters. Even if the inputs are identical, the output values may differ. For example, a counter has an output indicating that the preselection value has been reached. This output is set to 1 when the current value is equal to the preselection value.

EN

(*enable*) An optional block input. When enabled, an `ENO` output is set automatically.

If `EN` = 0, the block is not enabled; its internal program is not executed, and `ENO` is set to 0.

If `EN` = 1, the block's internal program is run and `ENO` is set to 1. If a runtime error is detected, `ENO` is set to 0.

If the `EN` input is not connected, it is set automatically to 1.

endianness

For multi-byte numbers, the big-endian and little-endian formats indicate the byte order in stored memory. In big-endian format, the most significant byte is stored in the lowest (first) address. In little-endian format, the least significant byte is stored in the lowest address. These examples show the contents of four memory bytes (address *x* ... address *x* + 3) for the multi-byte number 0A0B0C0D (h):
- big endian: *x* = 0A; *x* + 1 = 0B; *x* + 2 = 0C; *x* + 3 = 0D
- little endian: *x* = 0D; *x* + 1 = 0C; *x* + 2 = 0B; *x* + 3 = 0A

It is perhaps easier to remember these by their uncorrupted forms, which indicate that you first put the "big end in" (big endian) or "little end in" (little endian).

ENO

error notification The output associated with the optional input `EN`.

If `ENO` is set to 0 (either because `EN` = 0 or if a runtime error is detected):
- The status of the function block outputs remains the same as it was during the previous scanning cycle that executed correctly.
- The output(s) of the function, as well as the procedures, are set to 0.

Ethernet

A 10 Mb/s, 100 Mb/s, or 1 Gb/s, CSMA/CD, frame-based LAN that can run over copper twisted pair or fiber optic cable, or wireless. The IEEE standard 802.3 defines the rules for configuring a wired Ethernet network; the IEEE standard 802.11 defines the rules for configuring a wireless Ethernet network. Common forms include 10BASE-T, 100BASE-TX, and 1000BASE-T, which can utilize category 5e copper twisted pair cables and RJ45 modular connectors.

EtherNet/IP™

A network communication protocol for industrial automation applications that combines the standard internet transmission protocols of TCP/IP and UDP with the application layer common industrial protocol (CIP) to support both high speed data exchange and industrial control. EtherNet/IP employs electronic data sheets (EDS) to classify each network device and its functionality.

explicit messaging

TCP/IP-based messaging for Modbus TCP and EtherNet/IP. It is used for point-to-point, client/server messages that include both data—typically unscheduled information between a client and a server—and routing information. In EtherNet/IP, explicit messaging is considered class 3 type messaging, and can be connection-based or connectionless.

explicit messaging client

(*explicit messaging client class*) The device class defined by the ODVA for EtherNet/IP nodes that only support explicit messaging as a client. HMI and SCADA systems are common examples of this device class.

F

FBD

(*function block diagram*) A graphical programming language that works like a flowchart. By adding simple logical blocks (`AND`, `OR`, etc.), each function or function block in the program is represented in this graphical format. For each block, the inputs are on the left and the outputs on the right. Block outputs can be linked to inputs of other blocks in order to create complex expressions.

FDR

(*faulty device replacement*) A service that uses configuration software to replace an inoperable device.

FDT

(*field device tool*) The technology that harmonizes communication between field devices and the system host.

FTP

(*file transfer protocol*) A protocol that copies a file from one host to another over a TCP/IP-based network, such as the internet. FTP uses a client-server architecture as well as separate control and data connections between the client and server.

full duplex

The ability of 2 networked devices to independently and simultaneously communicate with each other in both directions.

G

gateway

A device that interconnects 2 different networks — sometimes with different network protocols. When used to connect networks based on different protocols, a gateway converts a datagram from one protocol stack into the other. When used to connect 2 IP-based networks, a gateway (also called a router) has 2 separate IP addresses – one on each network.

global data

Global data provides the automatic exchange of data variables for the coordination of PLC applications.

H

harsh environment

Resistance to hydrocarbons, industrial oils, detergents and solder chips. Relative humidity up to 100%, saline atmosphere, significant temperature variations, operating temperature between - 10° C and + 70° C, or in mobile installations.

high-capacity daisy chain loop

Often referred to as HCDCL, a high-capacity daisy chain loop uses DRSs to extend the distance between remote I/O drops or connect sub-rings (containing remote I/O drops or distributed I/O devices) and/or distributed I/O clouds to the Ethernet remote I/O network.

Hot Standby

A high-availability Quantum control system with a second (standby) PLC that maintains up-to-date system status. If the primary PLC becomes inoperable, the standby PLC takes control of the system.

HTTP

(*hypertext transfer protocol*) A networking protocol for distributed and collaborative information systems. HTTP is the basis of data communication for the web.

I

I/O scanning

Continuously polling the I/O modules to collect data and status, event, and diagnostics information. This process monitors inputs and controls outputs.

IEC 61131-3

International standard: programmable logic controllers

Part 3: programming languages

IGMP

(internet group management protocol) This internet standard for multicasting allows a host to subscribe to a particular multicast group.

IL

(*instruction list*) A series of basic instructions similar to assembly language used to program processors. Each instruction is made up of an instruction code and an operand.

implicit messaging

UDP/IP-based class 1 connected messaging for EtherNet/IP. Implicit messaging maintains an open connection for the scheduled transfer of control data between a producer and consumer. Because an open connection is maintained, each message contains primarily data — without the overhead of object information — and a connection identifier.

INT

(*integer*) (encoded in 16 bits) The upper/lower limits are as follows: -(2 to the power of 15) to (2 to the power of 15) - 1.

Example:

`-32768`, `32767`, `2#1111110001001001`, `16#9FA4`.

inter-controller network

An Ethernet-based network that is part of the control network, and provides data exchange between controllers and engineering tools (programming, asset management system (AMS)).

interlink port

An Ethernet port on Ethernet remote I/O modules allowing direct connection of distributed I/O modules to the remote I/O network.

IP address

The 32-bit identifier — consisting of both a network address and a host address — assigned to a device connected to a TCP/IP network.

isolated distributed I/O network

An Ethernet-based network containing distributed I/O devices that do not participate in an Ethernet remote I/O network.

J

jitter

Jitter is the time variation in the delivery of an Ethernet packet, caused by packet queuing along its network travel path. Jitter can be reduced to predictable amounts by applying packet handling policies—e.g. quality of service (QoS)—that grant priority to the packets of a specified type (e.g. remote I/O data packets) over other packet types.

L

LD

(*ladder diagram*) A programming language that represents instructions to be executed as graphical diagrams very similar to electrical diagrams (contacts, coils, etc.).

legacy remote I/O

A Quantum remote I/O system using coaxial cabling and terminators.

literal value of an integer

A value used to enter integer values in the decimal system. Values may be preceded by the "+" and "-" signs. Underscore signs (_) separating numbers are not significant.

Example:

`-12, 0, 123_456, +986`

local rack

A Quantum rack containing the controller, a power supply, and an Ethernet remote I/O head module. A local rack consists of 1 or 2 racks — the main rack (containing the remote I/O head module) and an optional extended rack. A Quantum Ethernet remote I/O network requires 1 local rack on the main ring.

local slave

A functionality offered by Schneider Electric EtherNet/IP communication modules that allows a scanner to take the role of an adapter. The local slave enables the module to publish data via implicit messaging connections. Local slave is typically used in peer-to-peer exchanges between PLCs.

M

MAST

A master processor task that is run through its programming software. The MAST task has 2 sections:
- IN: Inputs are copied to the IN section before execution of the MAST task.
- OUT: Outputs are copied to the OUT section after execution of the MAST task.

MIB

(*management information base*) A virtual database used for managing the objects in a communications network. See SNMP.

Modbus

An application-layer messaging protocol. Modbus provides client and server communications between devices connected on different types of buses or networks. Modbus offers many services specified by function codes.

Modbus/TCP

(*Modbus over TCP protocol*) A Modbus variant used for communications over TCP/IP networks.

multicast

A special form of broadcast where copies of the packet are delivered to only a specified subset of network destinations. Implicit messaging typically uses multicast format for communications in an EtherNet/IP network.

N

network

There are 2 meanings:

- In a ladder diagram:
  A set of interconnected graphic elements. The scope of a network is local, concerning the organizational unit (section) of the program containing the network.
- With expert communication modules:
  A set of stations that intercommunicate. The term *network* is also used to define a group interconnected graphic elements. This group then makes up part of a program that may comprise a group of networks.

NIM

(*network interface module*) A NIM resides in the first position on an STB island (leftmost on the physical setup). The NIM provides the interface between the I/O modules and the fieldbus master. It is the only module on the island that is fieldbus-dependent — a different NIM is available for each fieldbus.

NTP

(*network time protocol*) Protocol for synchronizing computer system clocks. The protocol uses a jitter buffer to resist the effects of variable latency.

O

O->T

(*originator to target*) See *originator* and *target*.

operation network

An Ethernet-based network containing operator tools (SCADA, client PC, printers, batch tools, EMS, etc.). PLCs are connected directly or through routing of the inter-controller network. This network is part of the control network.

originator

In EtherNet/IP, a device is considered the originator when it initiates a CIP connection for implicit or explicit messaging communications or when it initiates a message request for un-connected explicit messaging.

OS Loader

Firmware upgrade tool for Quantum hardware.

P

PLC

programmable logic controller. The PLC is the brain of an industrial manufacturing process. It automates a process as opposed to relay control systems. PLCs are computers suited to survive the harsh conditions of the industrial environment.

port 502

Port 502 of the TCP/IP stack is the well-known port that is reserved for Modbus communications.

port mirroring

In this mode, data traffic that is related to the source port on a network switch is copied to another destination port. This allows a connected management tool to monitor and analyze the traffic.

NOTE: In port mirroring mode, the SERVICE port acts like a read-only port. That is, you cannot access devices (ping, connection to Unity Pro, etc.) through the SERVICE port on the 140 CRP 312 00 and 140 CRA 312 00.

Q

QoS

(*quality of service*) The practice of assigning different priorities to traffic types for the purpose of regulating data flow on the network. In an industrial network, QoS is used to provide a predictable level of network performance.

Quantum Ethernet I/O device

These devices in Quantum Ethernet I/O systems provide automatic network recovery and deterministic remote I/O performance. The time it takes to resolve a remote I/O logic scan can be calculated, and the system can recover quickly from a communication disruption. Quantum Ethernet I/O devices include:
- local rack (with an Ethernet remote I/O head module)
- remote I/O drop (with an Ethernet adapter module)
- DRS pre-configured switch

R

rack optimized connection

Data from multiple I/O modules consolidated in a single data packet to be presented to the scanner in an implicit message in an EtherNet/IP network.

remote I/O drop

One of the 3 types of remote I/O devices in an Ethernet remote I/O network. A remote I/O drop is a Quantum rack of I/O modules that are connected to an Ethernet remote I/O network and managed by an Ethernet remote adapter module. A drop can be a single rack or a rack with an extension rack.

remote I/O main ring

The main ring of an Ethernet remote I/O network. The ring contains remote I/O devices and a local rack (containing a controller, a power supply module, and an Ethernet remote I/O head module).

remote I/O network

An Ethernet-based network that contains 1 standalone PLC or one Hot Standby system and remote I/O devices. There are 3 types of remote I/O devices: a local rack, a remote I/O drop, and a ConneXium extended dual-ring switch (DRS). Distributed I/O devices may also participate in a remote I/O network via connection to DRSs.

RPI

(requested packet interval) The time period between cyclic data transmissions requested by the scanner. EtherNet/IP devices publish data at the rate specified by the RPI assigned to them by the scanner, and they receive message requests from the scanner at each RPI.

RSTP

(*rapid spanning tree protocol*) A protocol that allows a network design to include spare (redundant) links to provide automatic backup paths if an active link stops working, without the need for loops or manual enabling/disabling of backup links.

S

scanner

The originator of I/O connection requests for implicit messaging in EtherNet/IP, and message requests for Modbus TCP.

scanner class device

An EtherNet/IP node capable of originating exchanges of I/O with other nodes in the network.

service port

A dedicated Ethernet port on the Quantum Ethernet remote I/O modules. The port may support 3 major functions (depending on the module type):
- port mirroring — for diagnostic use
- access — for connecting HMI/Unity Pro/ConneXview to the PLC
- extended — to extend the device network to another subnet
- disabled — disables the port, no traffic is forwarded in this mode

SFC

(*sequential function chart*) An IEC programming language that graphically represents, in a structured manner, the operation of a sequential PLC. This graphical description of the PLC's sequential behavior and of the various resulting situations is created using simple graphic symbols.

simple daisy chain loop

A daisy chain loop that contains remote I/O devices only (no switches or distributed I/O devices). This topology consists of a local rack (containing a remote I/O head module), and 1 or more remote I/O drops (each drop containing a remote I/O adapter module).

SNMP

(*simple network management protocol*) Protocol used in network management systems to monitor network-attached devices for events. The protocol is part of the internet protocol suite (IP) as defined by the internet engineering task force (IETF), which consists of network management guidelines, including an application layer protocol, a database schema, and a set of data objects.

SNTP

(*simple network time protocol*) See *NTP*.

SOE

(*sequence of events*) The process of determining the order of events in an industrial system and correlating those events to a real-time clock.

ST

(*structured text*) A structured, developed language similar to computer programming languages. It can be used to organize a series of instructions.

sub-ring

An Ethernet-based network with a loop attached to the main ring, via a DRS. A sub-ring may contain either remote I/O or distributed I/O devices.

subnet mask

The 32-bit value used to hide (or mask) the network portion of the IP address and thereby reveal the host address of a device on a network using the IP protocol.

switch

A multi-port device used to segment the network and limit the likelihood of collisions. Packets are filtered or forwarded based upon their source and destination addresses. Switches are capable of full-duplex operation and provide full network bandwidth to each port. A switch can have different input/output speeds (for example, 10, 100 or 1000 Mb/s). Switches are considered OSI layer 2 (data link layer) devices.

T

T->O

(*target to originator*) See *target* and *originator*.

target

In EtherNet/IP, a device that is the recipient of a connection request for implicit or explicit messaging communications, or when it is the recipient of a message request for un-connected explicit messaging.

TCP

(*transmission control protocol*) A key protocol of the internet protocol suite that supports connection-oriented communications, by establishing the connection necessary to transmit an ordered sequence of data over the same communication path.

TCP/IP

Also known as *internet protocol suite*, TCP/IP is a collection of protocols used to conduct transactions on a network. The suite takes its name from 2 commonly used protocols: transmission control protocol and internet protocol. TCP/IP is a connection-oriented protocol that is used by Modbus TCP and EtherNet/IP for explicit messaging.

TOD

(*time of day*) The `TOD` type, encoded in BCD in a 32-bit format, contains the following information:
- the hour encoded in an 8-bit field
- the minutes encoded in an 8-bit field
- the seconds encoded in an 8-bit field

NOTE: The 8 least significant bits are not used.

The TOD type is entered as follows: **`TOD#`**`<Hour>`**`:`**`<Minutes>`**`:`**`<Seconds>`

This table shows the upper/lower limits of each field:

Field	Limits	Comment
Hour	[00,23]	The leading 0 is displayed; it can be omitted during data entry.
Minute	[00,59]	The leading 0 is displayed; it can be omitted during data entry.
Second	[00,59]	The leading 0 is displayed; it can be omitted during data entry.

Example: `TOD#23:59:45`.

TR

(*transparent ready*) Web-enabled power distribution equipment, including medium- and low-voltage switch gear, switchboards, panel boards, motor control centers, and unit substations. Transparent Ready equipment allows you to access metering and equipment status from any PC on the network, using a standard web browser.

trap

An event directed by an SNMP agent that indicates one of the following:
- a change has occurred in the status of an agent
- an unauthorized SNMP manager device has attempted to get data from, or change data on, an SNMP agent

U

UDP

(*user datagram protocol*) A transport layer protocol that supports connectionless communications. Applications running on networked nodes can use UDP to send datagrams to one another. UDP does not always deliver datagrams as reliable or ordered as those delivered by TCP. However, by avoiding the overhead required for TCP, UDP is faster. UDP may be the preferred protocol for time-sensitive applications, where dropped datagrams are preferable to delayed datagrams. UDP is the primary transport for implicit messaging in EtherNet/IP.

V

variable

Memory entity of type `BOOL`, `WORD`, `DWORD`, etc., whose contents can be modified by the program currently running.

Index